Guia de Uso - Vehicle Rental API

v1.0.0

Sumário

1 Primeiras requisições

- 1.1 Autenticação
- 1.2 Criando um usuário

2 Usuários e veículos padrão

- 2.1 Usuários default
- 2.2 Veículos default

3 Roles/papéis da aplicação

4 Endpoints e autorizações

- 4.1 Endpoints
- 4.2 Permitir uso dos endpoints sem autenticação JWT

5 Informações adicionais

- 5.1 Estratégia de persistência
- 5.2 Endpoints com suporte a query params
  - 5.2.1 GET /v1/users?id={}&email={}
  - 5.2.2 GET /v1/rentals?customerEmail={}&vehicleChassis={}

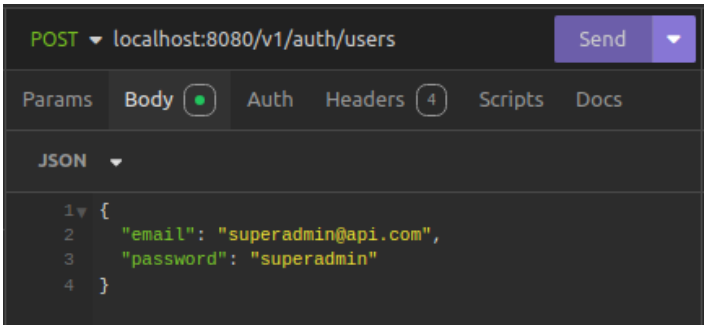
1 Primeiras requisições

Para pôr a aplicação com a API em execução pela primeira vez, siga o passo a passo disponibilizado no README da aplicação presente no repositório do GitHub do projeto.

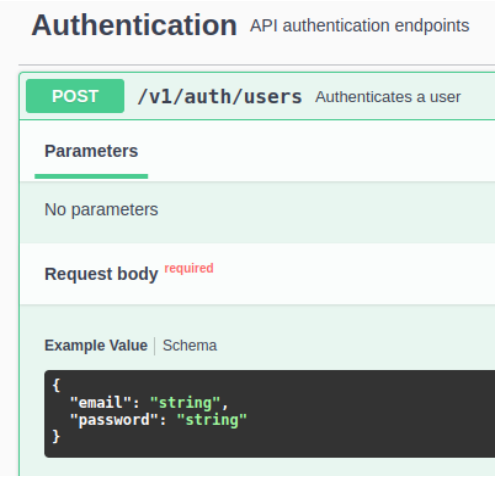
1.1 Autenticação

Todos os endpoints da aplicação são autenticados - com exceção das rotas de interface do Swagger e a própria rota **`/v1/auth/users`** utilizada para realizar as autenticações -, o que significa que para utilizar a API o cliente precisa ser um usuário cadastrado e realizar o login (gerar um JSON Web Token para uso nas próximas requisições) para realizar as requisições na API (uso dos endpoints).

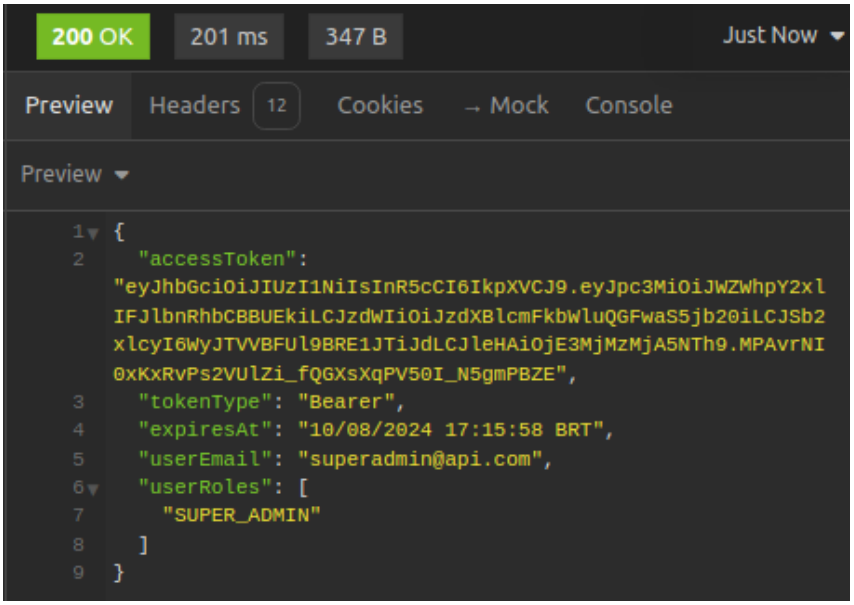
Para realizar o login forneça o **email** (não é o nome de usuário) e **senha** do usuário, por padrão 3 usuários são cadastrados assim que a aplicação é iniciada, nesse exemplo usaremos o usuário *superadmin*.



Informações sobre o endpoint da aplicação podem ser encontradas na interface de documentação do Swagger enquanto a aplicação estiver em execução:



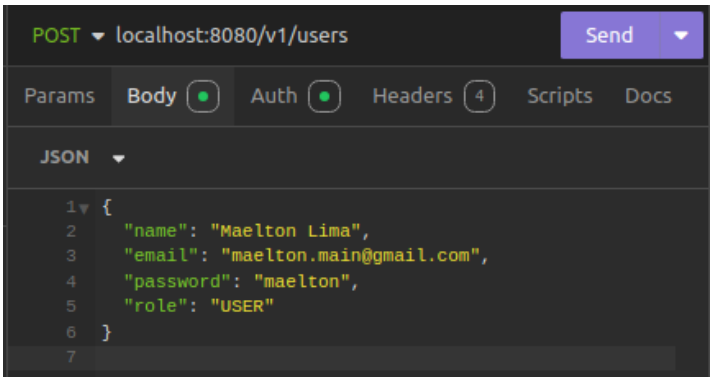
A resposta da requisição irá conter o JWT (JSON Web Token) e suas respectivas propriedades, como a data de expiração do token por exemplo:



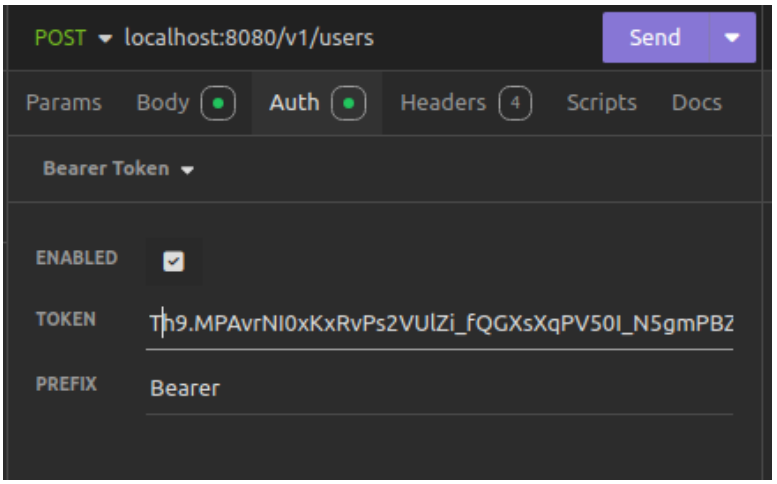
Agora basta copiar o valor do atributo *accessToken* e utilizá-lo nas próximas requisições, para isso passe-o como valor para o header *Authentication* como um Bearer Token.

1.2 Criando um usuário

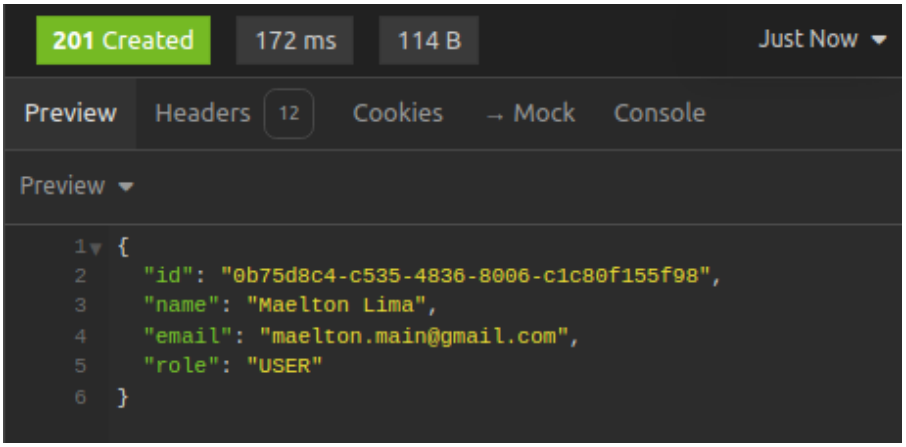
A partir do endpoint `**/v1/users**` crie uma requisição POST e forneça os dados do usuário que deseja cadastrar, a role do novo usuário deve ser obrigatoriamente especificada, podendo ser *SUPER_ADMIN*, *ADMIN* ou *USER*.



Apenas usuários administradores podem cadastrar novos usuários, sendo assim, certifique-se de fornecer um token de autorização gerado pelo login de um administrador. No nosso exemplo o usuário **Super Admin** possui a role *SUPER_ADMIN*, então não teremos problema, atualize o header de autorização do seu cliente e forneça o JWT antes de realizar a requisição.



Se a requisição for bem sucedida, você receberá uma resposta com o código 201 e os dados do novo usuário criado.


201 Created
172 ms
114 B
Just Now
Preview
Headers 12
Cookies
→ Mock
Console
Preview
{
"id": "0b75d8c4-c535-4836-8006-c1c80f155f98",
"name": "Maelton Lima",
"email": "maelton.main@gmail.com",
"role": "USER"
}

2 Usuários e veículos padrão

Tendo em vista que todas os endpoints da API são autenticados, incluindo o endpoint para criação de novos usuários, e que alguns endpoints precisam que o usuário possua autorização de uso - para atualizar ou deletar dados da aplicação o usuário precisa ser *SUPER_ADMIN* ou *ADMIN* por exemplo -, 3 usuários padrão são inseridos assim que a aplicação é iniciada, um para cada role/papel da aplicação (*SUPER_ADMIN, ADMIN, USER*).

A DatabaseInitializer é a classe que possui os métodos responsáveis pela configuração de inicialização do banco de dados, fique a vontade para verificar e modificá-la caso sinta necessidade.

2.1 Usuários default

NOME	EMAIL	SENHA	ROLE
Super Admin	superadmin@api.com	superadmin	SUPER_ADMIN
Admin	admin@api.com	admin	ADMIN
User	user@api.com	user	USER

2.2 Veículos default

ESPECIALIZAÇÃO	MODELO	CHASSIS
Car.class	Car Example Model	1
Motorcycle.class	Motorcycle Example Model	2

3 Roles/papéis da aplicação

A aplicação possui 3 roles/papéis: **SUPER_ADMIN**, **ADMIN** e **USER**. Elas foram utilizadas para garantir ou privar acessos a alguns endpoints da aplicação de acordo com a role de cada usuário e são necessárias para garantir o funcionamento da regra de negócios da aplicação.

4 Endpoints e autorizações

A configuração de autorizações da aplicação baseia-se nas roles de usuários e podem ser verificadas (e alteradas caso desejado) na classe SecurityConfiguration.

Em geral as configurações atuais visam atingir as regras de negócios atuais:

1. Apenas administradores podem criar, atualizar ou deletar veículos do sistema, usuários comuns não devem ter acesso a essas informações.
2. Apenas administradores podem visualizar a lista de usuários cadastrados no sistema, usuários comuns não devem ter acesso aos dados de outros usuários.
3. Apenas administradores podem **atualizar** ou **deletar** registros de locações/aluguel do sistema.
4. Usuários administradores podem locar/alugar veículos para **qualquer** usuário do sistema (eles inclusos).
5. Usuários **não** administradores podem locar/alugar veículos **apenas** para si mesmos.
6. **Todos** os usuários (administradores ou não) podem **visualizar** a lista de veículos cadastrados no sistema.

4.1 Endpoints

MÉTODO HTTP	RECURSO	ROLES AUTORIZADAS
POST	**/v1/cars**	SUPER_ADMIN, ADMIN
POST	**/v1/motorcycles**	SUPER_ADMIN, ADMIN
POST	**/v1/users**	SUPER_ADMIN, ADMIN
POST	**/v1/rentals**	SUPER_ADMIN, ADMIN
POST	**/v1/rentals/motorcycle**	SUPER_ADMIN, ADMIN, USER

POST	**/v1/rentals/car**	SUPER_ADMIN, ADMIN, USER
POST	**/v1/auth/users**	SUPER_ADMIN, ADMIN, USER
GET	**/v1/cars/{id}**	SUPER_ADMIN, ADMIN, USER
GET	**/v1/cars**	SUPER_ADMIN, ADMIN, USER
GET	**/v1/motorcycles/{id}**	SUPER_ADMIN, ADMIN, USER
GET	**/v1/motorcycles**	SUPER_ADMIN, ADMIN, USER
GET	**/v1/users/{id}**	SUPER_ADMIN, ADMIN
GET	**/v1/users**	SUPER_ADMIN, ADMIN
GET	**/v1/rentals/{id}**	SUPER_ADMIN, ADMIN, USER
GET	**/v1/rentals**	SUPER_ADMIN, ADMIN, USER
GET	**/v1/vehicles**	SUPER_ADMIN, ADMIN, USER
PUT	**/v1/cars/{id}**	SUPER_ADMIN, ADMIN
PUT	**/v1/motorcycles/{id}**	SUPER_ADMIN, ADMIN
PUT	**/v1/users/{id}**	SUPER_ADMIN, ADMIN
PUT	**/v1/rentals/{id}**	SUPER_ADMIN, ADMIN
DELETE	**/v1/cars/{id}**	SUPER_ADMIN, ADMIN
DELETE	**/v1/motorcycles/{id}**	SUPER_ADMIN, ADMIN
DELETE	**/v1/users/{id}**	SUPER_ADMIN, ADMIN
DELETE	**/v1/rentals/{id}**	SUPER_ADMIN, ADMIN

Para facilitar o uso e implementação da aplicação, usuários não administradores ainda conseguem visualizar todos os registros de locações, mesmo os que não são seus, mas isso pode e será alterado em versões futuras, de modo que usuários **não administradores** conseguirão visualizar apenas os próprios registros de locação/aluguel.

4.2 Permitir uso dos endpoints sem autenticação JWT

Para ser capaz de realizar todas as requisições sem a necessidade de fazer login (fornecer o token de autorização no cabeçalho Authorization da requisição) modifique as configurações presentes na classe SecurityConfiguration:

(O código presente na imagem pode ter sido alterado)

Caso queira permitir acesso a todos os endpoints sem necessidade de autorização, **para fins de teste**, modifique o código da imagem presente na classe SecurityConfiguration.

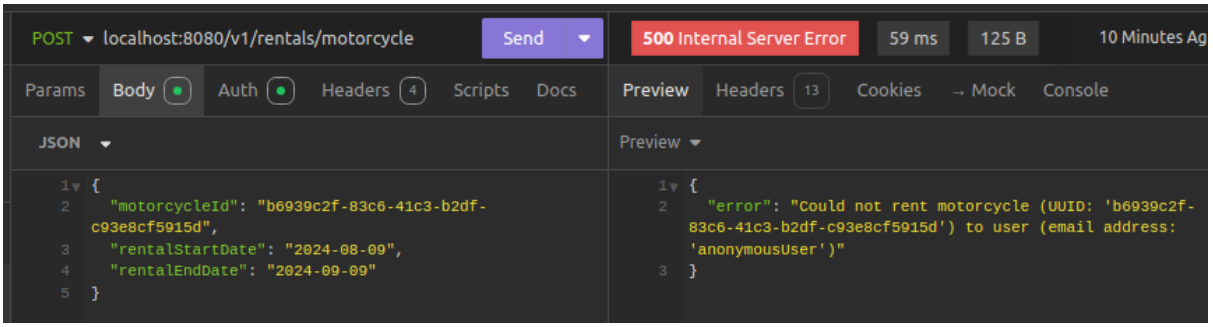
Para isso comente o código da imagem e adicione a linha de código **`.anyRequest().permitAll()`** logo após:

(O código presente na imagem pode ter sido alterado)

Após essas alterações você deverá ser capaz de realizar todas as requisições sem a necessidade de fornecer tokens de autorização. **Não esqueça** de

executar todos os comandos presentes no arquivo start.sh dentro do diretório local do projeto para gerar novos containers Docker com a aplicação atualizada.

Caso isso seja feito, os endpoints **`/v1/rentals/motorcycle`** e **`/v1/rentals/car`** não irão funcionar corretamente se o token do usuário não seja informado no momento da requisição. Esse comportamento acontece porque esses endpoints foram criados para que os usuários conseguissem locar veículos a partir do ID do veículo, ou seja, o ID do veículo é informado no corpo da requisição mas o ID do usuário não, a aplicação consegue "descobrir" quem é o cliente da locação através do token de autorização que o usuário informa no header Authorization da requisição - para visualizar as configurações de geração dos tokens da aplicação consulte a classe JWTService. Logo, se o token não é mais informado, a aplicação não consegue identificar quem é o locador do veículo e você possivelmente irá receber na response uma mensagem de erro parecida como essa:



Sendo assim, se for utilizar os endpoints **`/v1/rentals/motorcycle`** e **`/v1/rentals/car`** forneça os tokens de autenticação mesmo que os endpoints da API estejam liberados para acesso sem necessidade de autenticação.

É importante ressaltar que o funcionamento e uso dessa API foi idealizado para ocorrer em conjunto com as autenticações via JWT e é recomendado que todas as rotas sejam liberadas para uso sem necessidade de autenticação (**`.anyRequest().permitAll()`**) somente para fins de realização de testes.

5 Informações adicionais

5.1 Estratégia de persistência

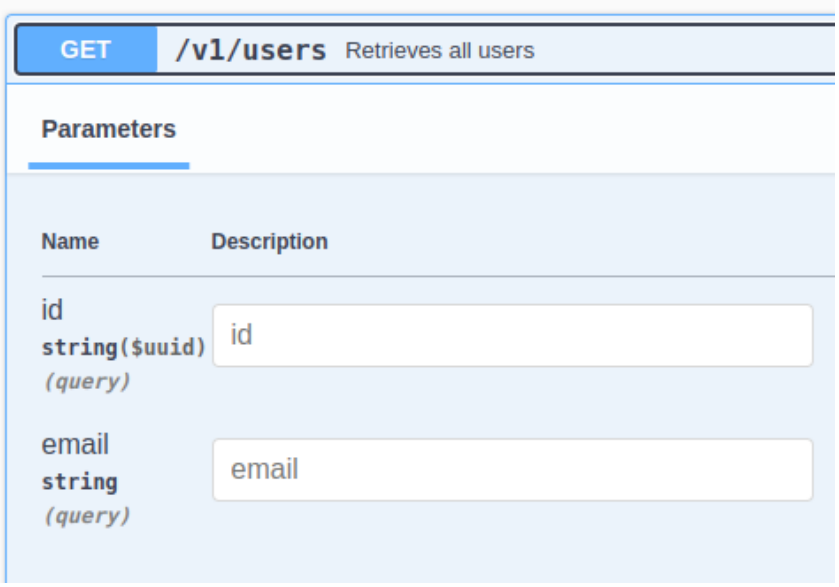
A persistência de dados foi realizada de forma automática em um banco de dados PostgreSQL a partir do uso da ferramenta Spring Data JPA e do Hibernate como ferramenta de ORM (Object-relational mapping).

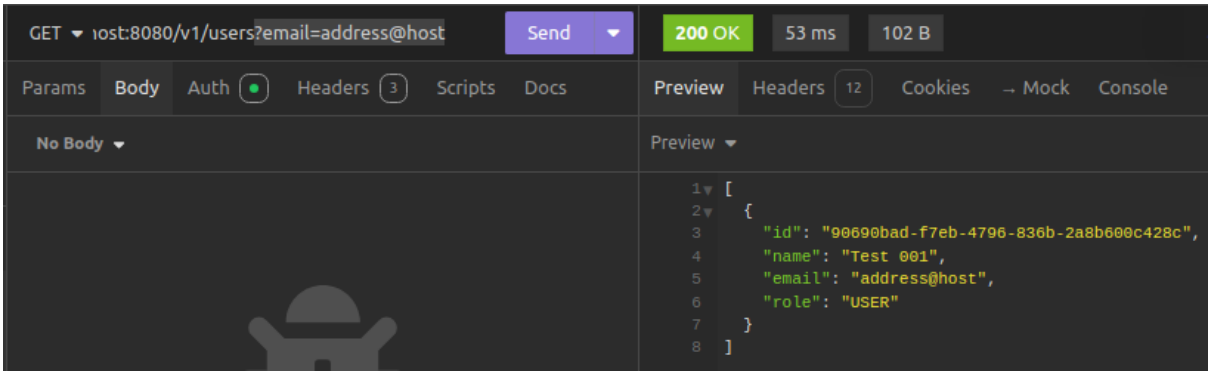
5.2 Endpoints com suporte a query params

Foram implementados query params **opcionais** em dois endpoints da aplicação, aproveite para fazer testes e realizar filtros de pesquisa, outros query params serão possivelmente adicionados nas próximas versões da aplicação.

5.2.1 GET /v1/users?id={}&email={}

É possível (opcionalmente) filtrar usuários por **id** e/ou **email** de forma isolada ou ao mesmo tempo - embora I**ds** e **emails** são únicos -, brinque com as possibilidades.





5.2.2 GET /v1/rentals?customerEmail={}&vehicleChassis={}

É possível (opcionalmente) filtrar registros de locações/aluguel por **email** do cliente e/ou **chassis** do veículo locado, de forma isolada ou ao mesmo tempo, brinque com as possibilidades.