Visão Global

Isso descreve os recursos que compõem a API REST do seguro residencial. Desenvolvida utilizando PHP 7 e o framework Laravel. A base de dados escolhida foi MySql.

Por padrão, todas as solicitações para a API: **http://127.0.0.1:8000/api/** recebem o **v1** da API REST, indicando qual a versão a ser utilizada.

O cabeçalho a ser envido deverá ser: **Accept: Application/json** para que a requisições retornadas sejam em formato JSON.

Obs: Na pasta “Documentacao” também é possível encontrar um arquivo com os endpoints para importar no Postman.

Esquema

Todo o acesso a API é feita e acessado a partir de **http://127.0.0.1:8000/api/.** Todos os dados são enviados e recebidos como JSON.

Autenticação

Existe apenas uma maneira de se autenticar através da API. As solicitações que requererem autenticação irão retornar **401 Unauthorized**, e informar se o token é inválido ou apenas expirou. O tempo para um token expirar é de 1 hora (60 minutos).

O Padrão de autenticação utilizado foi o JWT (Json Web Token), uma espécie de assinatura digital, que após o usuário está conectado, cada pedido que vier irá incluir o JWT, permitindo que o usuário continue acessando os serviços e recursos que foram liberados pelo token.

Essa autenticação deverá ser enviada no cabeçalho no formato:
Authorization: Bearer {TOKEN}

Pedido HTTP

Tipo	URL
POST	http://127.0.0.1:8000/api/v1/login

Parâmetros

Tipo	Campo	Valor
string	email	joao@gmail.com
string	password	12345678

Resposta

Se o usuário estiver cadastrado no base de dados irá retornar um token para ser utilizado no acesso e serviços que o usuário tiver permissão.

{"token":"eyJ0eXAiOiJKV1QiLCJhbGciOiJIUzI1NiJ9.eyJpc3MiOiJodHRwOlwvXC8xMjcuMC4wLjE6ODAwMFwvYXBpXC92MVwvbG9naW4iLCJpYXQiOjE1NzU4MjgyMzMsImV4cCI6MTU3NTgzMTgzMywibmJmIjoxNTc1ODI4MjMzLCJqdGkiOiJUbEhCUG9lV2xKT1NrUVllIiwic3ViIjo0LCJwcnYiOiI4N2UwYWYxZWY5ZmQxNTgxMmZkZWM5NzE1M2ExNGUwYjA0NzU0NmFhIn0.nmHSWtwS8FZ_9Oob7t2a35gcBGDJr0zW9BfVTTHNQFs"}

Se o usuário não estiver cadastrado irá retornar erro 404

{"message":"Unauthorized","errors":[]}

Usuários

Manipulação dos usuários que já estão cadastrado ou irão se cadastrar.

- Listar Usuários

Esse endpoint retorna informações sobre quais o usuários estão cadastrado no base de dados.

Pedido HTTP

Tipo	URL
GET	http://127.0.0.1:8000/api/v1/users

Resposta

Se existir algum usuários cadastrado irá retornar uma lista paginada com todos os usuários.


```
{
    "current_page": 1,
    "data": [
        {
            "id": 1,
            "nome": "João Pedro",
            "sobrenome": "Alves de Sousa",
            "cpf": "176.776.837-03",
            "email": "joao@gmail.com",
            "email_verified_at": null,
            "created_at": null,
            "updated_at": null
        }
]
}
```

- Listar Usuário Específico

Esse endpoint retorna informações sobre algum usuário em específico.

Pedido HTTP

Tipo	URL
GET	http://127.0.0.1:8000/api/v1/users/{id}

Resposta

Retorna as informações sobre o usuário buscado.


```
{
    "data": {
        "id": 1,
        "nome": "João Pedro",
        "sobrenome": "Alves de Sousa",
        "cpf": "176.776.837-03",
        "email": "joao@gmail.com",
        "email_verified_at": null,
        "created_at": null,
        "updated_at": null
    }
}
```


Se o id estiver incorreto irá retornar erro 401


```
{
    "message": "No query results for model [App\\User] 1",
    "errors": []
}
```

- Cadastrar Usuário

Esse endpoint cadastra um novo usuário.

Pedido HTTP

Tipo	URL
POST	http://127.0.0.1:8000/api/v1/users/

Parâmetros

Tipo	Campo	Formato
string	nome	
string	sobrenome	
string	cpf	000.000.000-00
string	email	exemplo@gmail.com
string	password	Mínimo 8 números

Obs: Todos os campos são obrigatórios.

Resposta

Se o cadastro ocorrer com sucesso, status 200


```
{
   "data": {
      "msg": "Usuário cadastrado com sucesso!"
   }
}
```

- Editar Usuário

Esse endpoint atualiza os dados do usuário escolhido.

Pedido HTTP

Tipo	URL
PUT	http://127.0.0.1:8000/api/v1/users/{id}

Parâmetros

Tipo	Campo	Formato
string	nome	
string	sobrenome	
string	cpf	000.000.000-00
string	email	exemplo@gmail.com
string	password	Mínimo 8 números

Resposta

Se a atualização ocorrer com sucesso, status 200


```
{
    "data": {
        "msg": "Usuário atualizado com sucesso!"
    }
}
```


Se a atualização estiver algum erro, status 422


```
{
    "message": "No query results for model [App\\User] 1",
    "errors": []
}
```

- Excluir Usuário

Esse endpoint exclui os dados de algum usuário.

Pedido HTTP

Tipo	URL
DELETE	http://127.0.0.1:8000/api/v1/users/{id}

Resposta

Se o usuário for valido.


```
{
    "data": {
        "msg": "Imovel removido com sucesso!"
    }
}
```


Se o id estiver incorreto irá retornar erro 401


```
{
    "message": "No query results for model [App\\Imovel] {id}",
    "errors": []
}
```

Imóveis

Lista todos os imóveis associados ao usuário autenticado.

- Listar Imóveis

Esse endpoint retorna informações sobre quais o imóveis associados ao usuário.

Pedido HTTP

Tipo	URL
GET	http://127.0.0.1:8000/api/v1/imovel/

Resposta

Se o usuário possuir algum imóvel em seu nome irá retornar uma lista paginada em 10 itens por página.


```
{
   "current_page": 1,
   "data": [
      {
         "id": 2,
         "id_usuario": 4,
         "logradouro": "Rua Esmeralda",
         "bairro": "Jardim Catarina",
         "municipio": "São Gonçalo",
         "estado": "Rio de Janeiro",
         "cep": "24716606",
         "tipo_imovel": "casa",
         "created_at": "2019-12-08 15:34:39",
         "updated_at": "2019-12-08 15:34:39"
      },
   ]
}
```

Se o usuário não estiver imóveis em seu nome será retornado a paginação vazia


```
{
    "current_page": 1,
    "data": [],
    "first_page_url": "http://127.0.0.1:8000/api/v1/imovel?page=1",
    "from": null,
    "last_page": 1,
    "last_page_url": "http://127.0.0.1:8000/api/v1/imovel?page=1",
    "next_page_url": null,
    "path": "http://127.0.0.1:8000/api/v1/imovel",
    "per_page": 10,
    "prev_page_url": null,
    "to": null,
    "total": 0
}
```

- Filtrar Lista de Imóvel

Esse endpoint retorna informações sobre quais o imóveis associados ao usuário, porém podendo realizar filtros.

Pedido HTTP

Tipo	URL
GET	http://127.0.0.1:8000/api/v1/busca?coditions=**{Paramentro}:=:{Valor}**

Parâmetros

Nome	Formato
logradouro	Rua Diamante
bairro	Jardim Catarina
cep	24716-606
tipo_imovel	casa ou apartamento

Exemplo 1: http://127.0.0.1:8000/api/v1/busca?coditions=logradouro:=:Rua Diamante

Exemplo 2: http://127.0.0.1:8000/api/v1/busca?coditions=logradouro:=:Rua Diamante%26bairro:=:Santa Luzia

Resposta


```
{
   "data": {
      "current_page": 1,
      "data": [
         {
            "id": 5,
            "id_usuario": 3,
            "logradouro": "Rua Diamante",
            "bairro": "Estrela do Norte",
            "municipio": "São Gonçalo",
            "estado": "Rio de Janeiro",
```

```
            "cep": "24715-505",
            "tipo_imovel": "casa",
            "created_at": "2019-12-14 00:00:00",
            "updated_at": "2019-12-07 00:00:00"
        }
    }
}
```

- Filtrar Imóvel pelo CPF

Esse endpoint retorna informações sobre quais o imóveis associados ao usuário, validado pelo CPF.

Pedido HTTP

Tipo	URL
POST	http://127.0.0.1:8000/api/v1/users/buscaImoveis

Parâmetros

Nome	Formato
cpf	000.000.000-00

Resposta


```
{
"data": [
        {
            "id": 1,
            "nome": "João Pedro",
            "sobrenome": "Alves de Sousa",
            "cpf": "176.776.837-03",
            "email": "joao@gmail.com",
            "id_usuario": 1,
            "logradouro": "Rua Expedicionario Aquiles Brasil",
            "bairro": "Jardim Catarina",
            "municipio": "São Gonçalo",
            "estado": "Rio de Janeiro",
            "cep": "24716-606",
            "tipo_imovel": "casa"
        }
}
```

- Listar Imóvel Específico

Esse endpoint retorna informações sobre algum imóvel em específico relacionado ao usuário autenticado.

Pedido HTTP

Tipo	URL
GET	http://127.0.0.1:8000/api/v1/imovel/{id}

Resposta

Se o usuário possuir algum imóvel em seu nome irá retornar os dados


```
{
    "data": {
        "id": 2,
        "id_usuario": 4,
        "logradouro": "Rua Esmeralda",
        "bairro": "Jardim Catarina",
        "municipio": "São Gonçalo",
        "estado": "Rio de Janeiro",
        "cep": "24716606",
        "tipo_imovel": "casa",
        "created_at": "2019-12-08 15:34:39",
        "updated_at": "2019-12-08 15:34:39"
    }
}
```


Se o id estiver incorreto irá retornar erro 401


```
{
    "message": "No query results for model [App\\Imovel] 1",
    "errors": []
}
```

- Cadastrar Imóvel

Esse endpoint cadastra um novo imóvel relacionado ao usuário cadastrado.

Pedido HTTP

Tipo	URL
POST	http://127.0.0.1:8000/api/v1/imovel/

Parâmetros

Tipo	Campo	Formato
string	logradouro	
string	bairro	
string	municipio	
string	estado	
string	cep	00000-000
int	Tipo_imovel	Casa ou apartamento

Obs: A relação entre o usuário e o imóvel é feito automático pela API. E todos os campos são obrigatórios.

Resposta

Se o cadastro ocorrer com sucesso, status 200


```
{
   "data": {
      "msg": "Imovel cadastrado com sucesso!"
   }
}
```

Se o cadastro estiver algum erro, status 422


```
{
    "message": "The given data was invalid.",
    "errors": {
        "logradouro": [
            "The logradouro field is required."
        ]
    }
}
```

- Editar Imóvel

Esse endpoint atualiza os dados do imóvel escolhido.

Pedido HTTP

Tipo	URL
PUT	http://127.0.0.1:8000/api/v1/imovel/{id}

Parâmetros

Tipo	Campo	Formato
string	logradouro	
string	bairro	
string	municipio	
string	estado	
string	cep	00000-000
int	Tipo_imovel	0 ou 1

Resposta

Se a atualização ocorrer com sucesso, status 200


```
{
    "data": {
        "msg": "Imovel atualizado com sucesso!"
    }
}
```


Se a atualização estiver algum erro, status 422


```
{
    "message": "No query results for model [App\\Imovel] 1",
    "errors": []
}
```

- Excluir Imóvel

Esse endpoint exclui os dados de algum imóvel.

Pedido HTTP

Tipo	URL
DELETE	http://127.0.0.1:8000/api/v1/imovel/{id}

Resposta

Se o usuário tiver o imóvel em seu nome


```
{
    "data": {
        "msg": "Imovel removido com sucesso!"
    }
}
```


Se o id estiver incorreto irá retornar erro 401


```
{
    "message": "No query results for model [App\\Imovel] {id}",
    "errors": []
}
```