Documentação da API REST do PlanIT

Introdução

A API REST do PlanIT permite gerenciar eventos, localizações, participantes e usuários de forma eficiente. Os endpoints fornecem suporte para criação, leitura, atualização e exclusão (CRUD) de recursos. Esta documentação detalha todos os endpoints, suas funções, métodos HTTP, parâmetros e exemplos de respostas.

Endpoints da API

1. Eventos

Adicionar Evento

- URL: `/event/add`

- Método: `POST`

- Descrição: Cria um novo evento.

- Corpo da Requisição:

 -json

 {

  "title": "Reunião Semanal",

  "description": "Discussão sobre metas",

  "date": "2025-01-15T10:00:00",

  "photoUrl": "https://example.com/event1.jpg",

  "userId": 1,

  "locationId": 2

 }

 -

- Resposta de Sucesso:

 -json

 {

  "message": "Event created successfully"

 }

 -

Buscar Eventos por Usuário

- URL: `/event/user/{userId}`

- Método: `GET`

- Descrição: Retorna todos os eventos criados por um usuário.

- Resposta de Sucesso:

-json

```
[
  {
    "id": 1,
    "title": "Reunião Semanal",
    "description": "Discussão sobre metas",
    "date": "2025-01-15T10:00:00",
    "photoUrl": "https://example.com/event1.jpg",
    "locationId": 2
  }
]
```

-

Buscar Todos os Eventos

- URL: `/event/all`

- Método: `GET`

- Descrição: Retorna todos os eventos registrados no sistema.

- Resposta de Sucesso:

-json

```
[
 {
  "id": 1,
  "title": "Reunião Semanal",
  "description": "Discussão sobre metas",
  "date": "2025-01-15T10:00:00",
  "photoUrl": "https://example.com/event1.jpg",
  "locationId": 2
 }
]
```


-

Deletar Evento

- URL: `/event/delete/{eventId}`

- Método: `DELETE`

- Descrição: Exclui um evento baseado no ID fornecido.

- Resposta de Sucesso:

 -

 HTTP 200 OK

 "Event successfully deleted"

 -

---

2. Localizações

Adicionar Localização

- URL: `/location/add`

- Método: `POST`

- Descrição: Adiciona uma nova localização ao sistema.

- Corpo da Requisição:

 -json

 {

  "address": "Rua das Flores, 123",

  "latitude": -23.550520,

  "longitude": -46.633308

 }

 -

- Resposta de Sucesso:

 -json

 {

  "id": 1,

  "address": "Rua das Flores, 123",

  "latitude": -23.550520,

  "longitude": -46.633308

 }

 -

Buscar Todas as Localizações

- URL: `/location/all`

- Método: `GET`

- Descrição: Retorna todas as localizações cadastradas.

- Resposta de Sucesso:

-json


```
[
  {
    "id": 1,
    "address": "Rua das Flores, 123",
    "latitude": -23.550520,
    "longitude": -46.633308
  }
]
```


-

---

3. Participantes

Adicionar Participante

- URL: `/participant/add`

- Método: `POST`

- Descrição: Adiciona um novo participante a um evento.

- Parâmetros da Requisição:

  - `eventId`: ID do evento (ex.: 1).

  - `userId`: ID do usuário (ex.: 2).

  - `status`: Status do participante (ex.: "confirmed").

- Resposta de Sucesso:

  -json

  {

    "id": 1,

    "userId": 2,

    "eventId": 1,

    "status": "confirmed"

  }

  -

Buscar Participantes por Evento

- URL: `/participant/event/{eventId}`

- Método: `GET`

- Descrição: Retorna todos os participantes associados a um evento.

- Resposta de Sucesso:

-json


```
[
  {
    "id": 1,
    "userId": 2,
    "userName": "Maria Oliveira",
    "eventId": 1,
    "eventTitle": "Reunião Semanal",
    "status": "confirmed"
  }
]
```


-

4. Usuários

Registrar Usuário

- URL: `/user/register`

- Método: `POST`

- Descrição: Registra um novo usuário no sistema.

- Corpo da Requisição:

 -json

 {

  "name": "João Silva",

  "email": "joao.silva@example.com",

  "password": "senha123"

 }

 -

- Resposta de Sucesso:

 -json

 {

  "id": 1,

  "name": "João Silva",

  "email": "joao.silva@example.com"

 }

 -

Login

- URL: `/user/login`

- Método: `POST`

- Descrição: Realiza o login de um usuário no sistema.

- Corpo da Requisição:

 -json

 {

  "email": "joao.silva@example.com",

  "password": "senha123"

 }

 -

- Resposta de Sucesso:

 -json

 {

  "id": 1,

  "name": "João Silva",

  "email": "joao.silva@example.com"

 }

 -