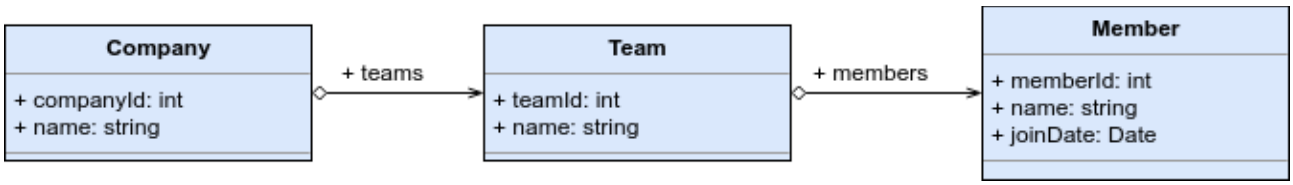
1 Estructura



/authorize [POST]
/companies [GET|POST]
/companies/companyId [GET|PUT]
/companies/companyId/teams [GET|POST]
/companies/companyId/teams/teamId [GET|PUT|DELETE]
/companies/companyId/teams/teamId/members [GET|POST]
/companies/companyId/teams/teamId/members/memberId [GET|PUT|DELETE]

2 Convenciones

Convenciones de **formato**:

- Sólo aceptamos formato JSON.
- Las respuestas sólo pueden contener los siguientes elementos:
  - success: obligatorio.
  - error: sólo si success == false.
  - data: según la petición, puede contener los datos de un recurso, o una lista de recursos.
- No se especifica el SQL de las tablas a propósito. Algunos frameworks puede generar el modelo de datos a partir de clases y los detalles de las tablas podrían variar.

Convenciones para **simplificar**.

- Utilizaremos un único error en lugar de una lista de errores.
- No validaremos el cuerpo de las peticiones.
- La API es pública y cualquiera puede leer y escribir.
- Un miembro sólo puede pertenecer a un equipo.
- No soportamos peticiones PATCH ya que aumentaría la complejidad considerablemente (ver https://tools.ietf.org/html/rfc6902 y https://tools.ietf.org/html/rfc6901).
- Ignoraremos el contenido de las cabeceras HTTP.

3 Recursos

3.1 Company

3.1.1 /companies

GET devolverá la lista de empresas:
URL: /companies
Method: GET

```
{
    "success": true,
    "data": [
        {
            "companyId": 1,
            "name": "atSistemas",
            "teams": "/companies/1/teams"
        },
        {
            "companyId": 2,
            "name": "Google",
            "members": "/companies/2/teams"
        }
    ]
}
```


POST creará una nueva empresa:
URL: /companies
Method: POST
Body:

```
{
    "name": "Company name"
}
```

```
{
    "sucess": true,
    "data": {
        "companyId": 5,
        "company": "/companies/5"
    }
}
```


3.1.2 /companies/companyId

GET devolverá los datos de la empresa:
URL: /companies/companyId
Method: GET

```
{
    "sucess": true,
    "data": {
        "companyId": 1,
        "name": "atSistemas",
        "teams": "/companies/1"
    }
}
```


PUT sobreescribirá los datos del equipo:
URL: /companies/companyId
Method: PUT
Body:

```
{
    "name": "atSistemas",
}
```

```
{
    "success": true,
    "company": "/companies/1"
}
```

DELETE no será posible para empresas.

3.2 Team

3.2.1 /companies/companyId/teams

GET devolverá la lista de equipos:
URL: /companies/teams
Method: GET

```
{
    "success": true,
    "data": [
        {
            "teamId": 1,
            "name": "PHP team",
            "members": "/companies/1/teams/1/members"
        },
        {
            "teamId": 2,
            "name": "UX",
            "members": "/companies/1/teams/2/members"
        }
    ]
}
```

POST creará un nuevo equipo:
URL: /companies/teams
Method: POST
Body:

```
{
    "name": "Team name"
}
```

```
{
    "sucess": true,
    "data": {
        "teamId": 5,
        "team": "/companies/1/teams/5"
    }
}
```

3.2.2 /companies/companyId/teams/teamId

GET devolverá los datos del equipo:
URL: /companies/companyId/teams/teamId
Method: GET

```
{
    "sucess": true,
    "data": {
        "teamId": 1,
        "name": "PHP team",
        "members": "/companies/1/teams/1/members"
    }
}
```


PUT sobreescribirá los datos del equipo:
URL: /companies/companyId/teams/teamId
Method: PUT
Body:

```
{
    "name": "PHP team",
}
```

```
{
    "success": true,
    "team": "/companies/1/teams/5"
}
```


DELETE eliminará el equipo, sólo si está vacío:
URL: /companies/companyId/teams/teamId
Method: DELETE

```
{
    "success": true
}
```

3.3 Member

3.3.1 /companies/companyId/teams/teamId/members

GET devolverá la lista de miembros:
URL: /companies/companyId/teams/teamId/members
Method: GET

```
{
    "success": true,
    "data": [
        {
            "memberId": 35,
            "name": "Pepe",
            "joinDate": "2016-01-15"
        },
        {
            "memberId": 21,
            "name": "Adolfo",
            "joinDate": "2012-10-02"
        }
    ]
}
```


POST creará un nuevo miembro:
URL: /companies/companyId/teams/teamId/members
Method: POST
Body:


```
{
    "name": "Member name",
    "joinDate": "2017-05-01"
}
```


(joinDate será opcional, en caso de omitirse se utilizará la fecha actual)

```
{
    "success": true,
    "memberId": 44,
    "member": "/companies/1/teams/1/members/44"
}
```

3.3.2 /companies/companyId/teams/teamId/members/memberId

GET devolverá los datos del miembro:
URL: /companies/companyId/teams/teamId/members/memberId
Method: GET

```
{
    "memberId": 35
    "name": "Pepe",
    "joinDate": "2016-01-15"
}
```

PUT sobreescribirá los datos del miembro:
URL: /companies/companyId/teams/teamId/members
Method: PUT
Body:

```
{
    "name": "Pepe",
    "joinDate": "2016-01-15"
}
```

(joinDate será opcional, en caso de omitirse se utilizará la fecha actual)

```
{
    "success": true,
    "member": "/companies/1/teams/1/member/35"
}
```

DELETE eliminará el miembro:
URL: /companies/companyId/teams/teamId/members
Method: DELETE

```
{
    "success": true
}
```

4 Errores

Además del mensaje de error, la respuesta incluirá el código de estado HTTP oportuno (200 para peticiones con éxito).

Recursos no encontrados (404):
- Team #teamId not found.
- Team member #memberId not found.

Método no implementado (501):
- Method not supported.

Ejemplo:

```
{
    "success": false,
    "error": "Team #33 not found."
}
```