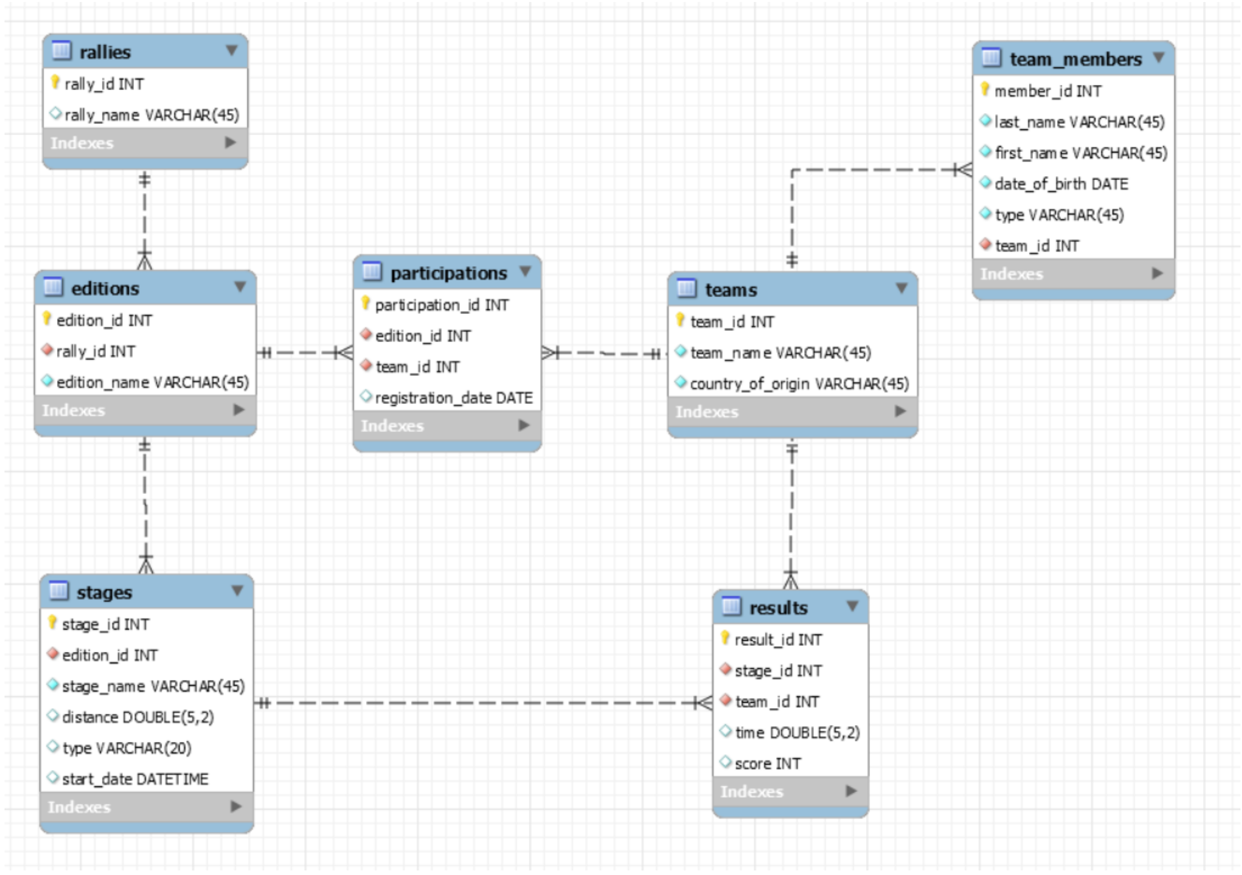
Rally Competition Java Project

Datorita interesului din ce in ce mai ridicat pentru competiile sportive de tip raliu, am descis sa implementez un model ce sa faciliteze utilizatorilor sa creeze noi competitii, sa vizualizeze detalii despre competitiile existente, sa le actualizeze, sa-si creeze echipe cu care sa participe la aceste competitii si unde, de asemenea, sa poata tine evidenta rezultatelor obtinute.

Aplicatia mea are la baza o serie de tabele ce permit manipularea datelor dupa cum am prezentat anterior. Diagrama conceptuala poate fi observata in imaginea urmatoare.



Astfel, avem entitatile:

- Rallies – retine id-ul raiului, precum si numele care este unic;
- Editions – retine editiile raliurilor, astfel o editie poate apartine unui singur raliu, iar un raliu poate avea mai multe editii;

- Stages – aceasta entitate retine informatii despre etapele asociate editiilor, astfel, ca si in cazul anterior, si aici se stabileste o relatie de tip 1:M, in sensul ca o etapa e asociata unei singure editii, in timp ce o editie are mai multe etape;
- Teams – retine informatii despre echipele definite in sistem, echipe ce pot participa la competitii;
- Team members – aceasta entitate retine informatii despre memebrii echipelor, un membru poate apartine unei singure echipe, combinatia de (last_name, first_name, date_of_birth) este privita ca o combinatie unica, iar valorile posibile pentru tipul unui membru sunt: driver,coDriver si staff;
- Participations – retine informatii despre participarea echipelor la diverse editii ale unor raliuri;
- Results – permite retinerea de informatii cu privile la rezultatele unei echipe la o anumita etapa, pe baza acestui tabel se poate calcula un clasament in ceea ce priveste o anumita editie a unui raliu.

Servicii

In continuare voi prezenta scopurile definite la nivelul fiecarui serviciu implementat in aplicatie.

RallyService se ocupa de:

- Administrarea adaugarii unui raliu cu sau fara editii si etape;
- Administrarea obtinerii editiilor unui raliu
- Administrarea modificarii numelui unui raliu
- Administrarea eliminarii unui raliu impreuna cu editiile si etapele aferente din sistem

EditionService se ocupa de:

- Administrarea adaugarii unei editii la un raliu cu sau fara etape
- Administrarea adaugarii de etape la o editie daca aceasta exista sau crearea editiei cu adaugarea etapelor la editia nou creata in sistem
- Administrarea obtinerii unei editii pe baza numelui si a numelui raliului caruia ii apartine
- Administrarea obtinerii tuturor editiilor unui raliu
- Administrarea obtinerii editiilor cu un nume dat, indiferent de raliuri

StageService se ocupa de:

- Administrarea adaugarii de etape la o editie existenta
- Administrarea obtinerii tuturor etapelor de la o anumita editie din cadrul unui anumit raliu pe baza numelor

TeamService se ocupa de:

- Administrarea adaugarii unei echipe in sistem cu sau fara membri
- Administrarea obtinerii unei echipe pe baza numelui
- Administrarea modificarii numelui unei echipe

TeamMemberService se ocupa de:

- Administrarea adaugarii unui nou membru la o echipa

ParticipationService se ocupa de:

- Administrarea inscrierii unei echipe la o anumita editie
- Administrarea anularii unei inscrieri a unei echipe la o anumita editie
- Administrarea obtinerii editiilor la care a participat o echipa
- Administrarea obtinerii echipelor ce au participat la o anumita editie

ResultService se ocupa de:

- Administrarea adaugarii unei rezultat obtinut de o echipa la o anumita etapa
- Administrarea actualizarii scorului unei echipe la un anumit rezultat
- Administrarea obtinerii unui clasament al echipelor la o anumita editie

Business Requirements

1. Adaugarea unui raliu in sistem – posibilitatea de a adauga un raliu(crearea unui raliu cu numele dat) sau crearea unui raliu cu intreaga structura de sub acesta(editii+etape)
2. Adaugarea unei editii la un raliu cu sau fara etape
3. Adaugarea unei liste de etape la un raliu deja existent
4. Modificarea numelui unui raliu
5. Obtinerea unui raliu cu intreaga structura (editii+etape) pe baza numelui
6. Eliminarea unui raliu din sistem impreuna cu intreaga structura a acestuia pe baza numelui
7. Obtinerea informatiilor despre o anumita editie a unui raliu pe baza numelor

8. Obtinerea informatiilor despre toate editiile cu un anumit nume
9. Obtinerea etapelor prezente intr-un anumit raliu la o anumita etapa pe baza numelor acestora
10. Adaugarea unei echipe in sistem cu sau fara membri
11. Obtinerea structurii unei echipe dupa numele echipei
12. Modificarea numelui unei echipe
13. Adaugarea unui membru la o anumita echipa pe baza numelui echipei
14. Inscrierea unei echipe la o anumita editie a unui raliu
15. Anularea inscrierii unei echipe la o editie a unui raliu
16. Obtinerea tuturor participarilor unei echipe (indiferent de editie)
17. Obtinerea echipelor participante la o anumita editie
18. Adaugarea rezultatului obtinut de o echipa la o etapa
19. Modificarea scorului unui anumit rezultat
20. Obtinerea rezultatelor de la o anumita editie sub forma de clasament

Select a definition default

Rally Competition API Documentation

`[ Base URL: localhost:8080/ ]`

http://localhost:8080/v2/api-docs

API Documentation for all available operations with rallies, editions, stages, teams, team members, team participations in rally editions and their stage results

Edition Edition Controller

POST `/edition` Create an edition

Creates a new Edition based on the information received in the request

Parameters Try it out

Name	Description
edition * required `object` *(body)*	Edition details **Example Value** Model

```
{
  "editionName": "string",
  "rallyName": "string",
  "stages": [
    {
      "distance": 0,
      "stageId": 0,
      "stageName": "string",
      "startDate": "2021-01-14T18:21:52.562Z",
      "type": "string"
    }
  ]
}
```

Parameter content type

application/json

Responses

Response content type */*

Code	Description
200	OK **Example Value** Model `{ "body": {}, "statusCode": "ACCEPTED", "statusCodeValue": 0 }`
201	The Edition was successfully created based on the received request
400	Validation error on the received request
401	Unauthorized
403	Forbidden
404	Not Found

```
{
  "body": {},
  "statusCode": "ACCEPTED",
  "statusCodeValue": 0
}
```

GET `/edition/{editionName}` Get editions

Get editions which have the name received from parameter

Parameters

Try it out

Name	Description

Name	Description
edition name * required string *(path)*	Name of target edition edition name - Name of target edition

Responses **Response content type** */*

Code	Description
200	The Editions were found
401	Unauthorized
403	Forbidden
404	Not Found

200 — **Example Value** Model

```
[
  {
    "editionName": "string",
    "rallyName": "string",
    "stages": [
      {
        "distance": 0,
        "stageId": 0,
        "stageName": "string",
        "startDate": "2021-01-14T18:23:03.991Z",
        "type": "string"
      }
    ]
  }
]
```

GET `/edition/{rallyName}/{editionName}` Get an edition

Get edition which have the specified name and belongs to the rally with name specified in the first parameter

Parameters

Try it out

Name	Description
edition name * required string *(path)*	Name of target edition edition name - Name of target edition
rally name * required string *(path)*	Name of target rally rally name - Name of target rally

Responses

Response content type */*

Code	Description
200	The Edition was found

Example Value Model

```
{
  "editionName": "string",
  "rallyName": "string",
  "stages": [
    {
      "distance": 0,
      "stageId": 0,
      "stageName": "string",
      "startDate": "2021-01-14T18:23:04.005Z",
      "type": "string"
    }
  ]
}
```

Code	Description
401	Unauthorized
403	Forbidden
404	Not Found

Participation Participation Controller

POST `/participation` Create a Participation

Enroll the current user's team in a rally edition

Parameters

Try it out

Name	Description
enroll * required `object` *(body)*	Edition and user details

Example Value Model

```
{
  "editionName": "string",
  "rallyName": "string",
  "teamMember": {
    "dateOfBirth": "2021-01-14",
    "firstName": "string",
    "lastName": "string",
    "type": "coDriver"
  }
}
```

Parameter content type

application/json

Responses

Response content type */*

Code	Description

Code	Description
200	OK **Example Value** Model `{ "body": {}, "statusCode": "ACCEPTED", "statusCodeValue": 0 }`
201	The team of the team member was successfully registered at rally edition received in the request
400	Validation error on the received request
401	Unauthorized
403	Forbidden
404	Not Found

Example value for 200, as printed:

```
{
  "body": {},
  "statusCode": "ACCEPTED",
  "statusCodeValue": 0
}
```

DELETE `/participation` Delete a Participation

Unregister the current user's team from a rally edition

Parameters Try it out

Name	Description

Name **Description**

enroll * required
`object`
(body)

Edition and user details

Example Value Model

```
{
  "editionName": "string",
  "rallyName": "string",
  "teamMember": {
    "dateOfBirth": "2021-01-14",
    "firstName": "string",
    "lastName": "string",
    "type": "coDriver"
  }
}
```

Parameter content type

application/json

Responses

Response content type */*

Code **Description**

200

The team of the team member was successfully unregistered at rally edition received in the request

Example Value Model

```
{
  "body": {},
  "statusCode": "ACCEPTED",
  "statusCodeValue": 0
}
```

204

No Content

401

Unauthorized

403

Forbidden

404

The team member or edition you are looking for wasn't found

GET **/participation/{teamName}** Get team participation

Get all the participation of a team regardless of edition and rally

Parameters

Try it out

Name	Description
teamName * required string *(path)*	Name of the team teamName - Name of the team

Responses

Response content type */*

Code	Description
200	The team was found **Example Value** Model
401	Unauthorized
403	Forbidden

```
[
  {
    "editionName": "string",
    "rallyName": "string",
    "stages": [
      {
        "distance": 0,
        "stageId": 0,
        "stageName": "string",
        "startDate": "2021-01-14T18:23:04.016Z",
        "type": "string"
      }
    ]
  }
]
```

Code	Description
404	The team wasn't found

GET `/participation/editionParticipations/{editionId}` Get participation on edition

Get all teams participating in the edition whose id is received as a parameter

Parameters

Try it out

Name	Description
editionId * required `integer($int32)` *(path)*	Id of the target Edition editionId - Id of the target Edition

Responses

Response content type */*

Code	Description
200	The edition was found **Example Value** Model

```
[
  {
    "countryOfOrigin": "string",
    "teamMembers": [
      {
        "dateOfBirth": "2021-01-14",
        "firstName": "string",
        "lastName": "string",
        "type": "coDriver"
      }
    ],
    "teamName": "string"
  }
]
```

Rally Rally Controller

Code Description

POST `/rally` Create a Rally

Creates a new Rally based on the information received in the request. It also adds Editions if exists

Parameters Try it out

Name	Description
rally * required `object` *(body)*	Rally details

Example Value Model

```
{
  "editions": [
    {
      "editionName": "string",
      "stages": [
        {
          "distance": 0,
          "stageId": 0,
          "stageName": "string",
          "startDate": "2021-01-14T18:22:11.220Z",
          "type": "string"
        }
      ]
    }
  ],
  "rallyId": 0,
  "rallyName": "string"
}
```

Parameter content type

application/json

Responses **Response content type** */*

Code	Description

Code	Description
200	OK **Example Value** Model `{ "body": {}, "statusCode": "ACCEPTED", "statusCodeValue": 0 }`
201	Rally was added
400	Validation error on the received request
401	Unauthorized
403	Forbidden
404	Not Found

```
{
  "body": {},
  "statusCode": "ACCEPTED",
  "statusCodeValue": 0
}
```

GET `/rally/{rallyName}` Get rally

Get rally by rally name which is unique

Parameters Try it out

Name	Description
rallyName * required `string` *(path)*	Name of target rally rallyName - Name of target rally

Responses

Response content type */*

Code **Description**

200

The rally was found

Example Value Model

```
[
  {
    "editionName": "string",
    "rallyName": "string",
    "stages": [
      {
        "distance": 0,
        "stageId": 0,
        "stageName": "string",
        "startDate": "2021-01-14T18:23:04.026Z",
        "type": "string"
      }
    ]
  }
]
```

Code	Description
401	Unauthorized
403	Forbidden
404	The rally wasn't found

DELETE `/rally/{rallyName}` Delete a rally

Delete a rally with the name received as parameter

Parameters

Try it out

Name	Description
rallyName * required string *(path)*	Name of target rally rallyName - Name of target rally

Responses

Response content type */*

Code	Description
200	OK **Example Value** Model `{ "body": {}, "statusCode": "ACCEPTED", "statusCodeValue": 0 }`
204	No Content
401	Unauthorized
403	Forbidden

```
{
  "body": {},
  "statusCode": "ACCEPTED",
  "statusCodeValue": 0
}
```

PUT `/rally/changeRallyName/{previousName}/{changedName}` Change rally name

Parameters

Try it out

Name	Description
newName * required `string` *(path)*	New name of rally newName - New name of rally
previousName * required `string` *(path)*	Previous name of rally previousName - Previous name of rally

Responses

Response content type */*

Code	Description
200	The name of the rally was changed **Example Value** Model
201	Created
401	Unauthorized
403	Forbidden
404	The rally wasn't found

```
{
  "body": {},
  "statusCode": "ACCEPTED",
  "statusCodeValue": 0
}
```

Result Result Controller

POST `/result` Create a Result

Create a result of team on a stage

Parameters

Try it out

Name	Description

Name **Description**

result * required
object
(body)

Result details

Example Value Model

```
{
  "score": 0,
  "stageId": 0,
  "teamId": 0,
  "teamName": "string",
  "time": 0
}
```

Parameter content type

application/json

Responses

Response content type */*

Code **Description**

200

The result was created

Example Value Model

```
{
  "body": {},
  "statusCode": "ACCEPTED",
  "statusCodeValue": 0
}
```

201

Created

400

Validation error on the received request

401

Unauthorized

403

Forbidden

404

Not Found

GET `/result/{editionId}` Get results

Get results of an edition

Parameters

Try it out

Name	Description
editionId * required `integer($int32)` *(path)*	Id of searched edition editionId - Id of searched edition

Responses

Response content type */*

Code	Description
200	Edition was found **Example Value** Model
401	Unauthorized
403	Forbidden
404	Edition wasn't found

Example for 200:

```
[
  {
    "countryOfOrigin": "string",
    "teamName": "string",
    "totalScore": 0
  }
]
```

PUT `/result/{resultId}/{score}` Update the score of a result

Update a score of a result if the resultId is found

Parameters

Try it out

Name	Description
resultId * required `integer($int32)` *(path)*	Id of target result resultId - Id of target result
score * required `integer($int32)` *(path)*	The value of the score to be added score - The value of the score to be added

Responses

Response content type */*

Code	Description
200	The score was updated **Example Value** Model
201	Created
401	Unauthorized
403	Forbidden

```
{
  "body": {},
  "statusCode": "ACCEPTED",
  "statusCodeValue": 0
}
```

Code	Description
404	The result wasn't found

Stage Stage Controller

POST /stage Add stages to an edition

Add stages to edition if it exists

Parameters

Try it out

Name	Description
edition * required object *(body)*	Edition with stages details

Example Value Model

```
{
  "editionName": "string",
  "rallyName": "string",
  "stages": [
    {
      "distance": 0,
      "stageId": 0,
      "stageName": "string",
      "startDate": "2021-01-14T18:22:27.099Z",
      "type": "string"
    }
  ]
}
```

Parameter content type

application/json

Responses

Response content type */*

Code **Description**

200

Stages were created

Example Value Model

```
{
  "body": {},
  "statusCode": "ACCEPTED",
  "statusCodeValue": 0
}
```

Code	Description
201	Created
400	Edition not found
401	Unauthorized
403	Forbidden
404	Not Found

GET `/stage/{rallyName}/{editionName}` Get stages

Get stages of a specified edition from a given rally

Parameters

Try it out

Name	Description
editionName * required `string` *(path)*	Name of target edition editionName - Name of target edition

Name	Description
rallyName * required string *(path)*	Name of target rally rallyName - Name of target rally

Responses **Response content type** */*

Code	Description
200	The stages were obtained
401	Unauthorized
403	Forbidden
404	Rally or edition not found

For code 200: **Example Value** Model

```
[
  {
    "distance": 0,
    "stageId": 0,
    "stageName": "string",
    "startDate": "2021-01-14T18:23:04.054Z",
    "type": "string"
  }
]
```

Team Team Controller

POST `/team` Create a team

Creates a team and if it has team members creates also the members

Parameters

Try it out

Name	Description
edition * required `object` *(body)*	Team and team members details **Example Value** Model

```
{
  "countryOfOrigin": "string",
  "teamId": 0,
  "teamMembers": [
    {
      "dateOfBirth": "2021-01-14",
      "firstName": "string",
      "lastName": "string",
      "teamMemberId": 0,
      "type": "coDriver"
    }
  ],
  "teamName": "string"
}
```

Parameter content type

application/json

Responses

Response content type */*

Code	Description
200	Team and team members were created **Example Value** Model

```
{
  "body": {},
  "statusCode": "ACCEPTED",
  "statusCodeValue": 0
}
```

Code	Description
201	Created
400	Validation error on the received request
401	Unauthorized
403	Forbidden
404	Not Found

GET `/team/{teamName}` Get team

Get team with team members by team name which is unique

Parameters

Try it out

Name	Description
teamName * required `string` *(path)*	Name of the team teamName - Name of the team

Responses

Response content type */*

Code	Description

Code **Description**

200

The information was get

Example Value Model

```
{
  "countryOfOrigin": "string",
  "teamMembers": [
    {
      "dateOfBirth": "2021-01-14",
      "firstName": "string",
      "lastName": "string",
      "type": "coDriver"
    }
  ],
  "teamName": "string"
}
```

Code	Description
204	Team wasn't found
401	Unauthorized
403	Forbidden
404	Not Found

PUT `/team/changeTeamName/{previousName}/{changedName}` Change team name

Parameters Try it out

Name	Description
newName * required `string` *(path)*	New name of rally newName - New name of rally
previousName * required `string` *(path)*	Previous name of team previousName - Previous name of team

Responses

Response content type */*

Code **Description**

200

The name of the team was changed

Example Value Model

```
{
  "body": {},
  "statusCode": "ACCEPTED",
  "statusCodeValue": 0
}
```

201

Created

401

Unauthorized

403

Forbidden

404

The team wasn't found

Team Member Team Member Controller

POST `/teamMember/{teamName}` Create a team member

Creates a new team member on the team with the name received by parameter

Parameters

Try it out

Name **Description**

teamMember * required
`object`
(body)

Team members details

Example Value Model

```
{
  "dateOfBirth": "2021-01-14",
  "firstName": "string",
  "lastName": "string",
  "team": {
    "countryOfOrigin": "string",
    "teamId": 0,
    "teamMembers": [
      null
    ],
    "teamName": "string"
  },
  "teamMemberId": 0,
  "type": "coDriver"
}
```

Parameter content type

application/json

teamName * required
`string`
(path)

Name of the team

teamName - Name of the team

Responses

Response content type */*

Code **Description**

200

Team member was created and added to the team

Example Value Model

```
{
  "body": {},
  "statusCode": "ACCEPTED",
  "statusCodeValue": 0
}
```

201

Created

Code	Description
400	Team wasn't found
401	Unauthorized
403	Forbidden
404	Not Found

Models

```
Edition    {
    description:
                          Entire structure of an Edition

    editionName*          string

                          editionName

    rally                 Rally   {...}

    stages                   [...]
}
```

```
Edition Dto   {
    description:
                          Required details needed to create a new Edition

    editionName*          string
    rallyName*            string
    stages
                             [...]
}
```

```
Enroll team to edition  {
    description:
                          Required details needed to register the team for the edition

    editionName*          string
    rallyName*            string
    teamMember*
                          Team Member Dto  {...}
}
```

```
Participation  {
    description:
                          Returned values for a participation

    countryOfOrigin       string
    teamName              string
    totalScore            integer($int32)
}
```

```
Rally  {
    description:
                          Required details needed to create a new Rally

    editions
                              [...]
    rallyId               integer($int64)
    rallyName*            string
}
```

```
ResponseEntity
```

```
Result Dto  {
    description:
                          Required details needed to create a new Result

    score*                integer($int64)
    stageId*              integer($int32)
    teamId                integer($int32)
    teamName*             string
    time*                 number($double)
}
```

Stage {

```
    description:
                             Entire structure of a Stage

    distance*                number($double)
                             minimum: 0
                             exclusiveMinimum: false
    edition                  Edition {...}
    stageId                  integer($int64)
    stageName*               string
    startDate*               string($date-time)
    type*                    string
}
```

Stage Dto {

```
    description:
                             Required details needed to create a new Stage

    distance*                number($double)
                             minimum: 0
                             exclusiveMinimum: false
    stageId                  integer($int64)
    stageName*               string
    startDate*               string($date-time)
    type*                    string
}
```

Team {

```
    description:
                             Entire structure of a Team

    countryOfOrigin*         string
    teamId                   integer($int64)
    teamMembers
                                 [...]
    teamName*                string
}
```

Team Dto {

```
    description:
                             Required details needed to create a new Team

    countryOfOrigin          string
    teamMembers
                                 [...]
    teamName                 string
}
```

```
Team Member    {
   description:
                          Entire structure of a Team Member

   dateOfBirth*           string($date)
   firstName*             string
   lastName*              string
   team
                          Team    {...}
   teamMemberId           integer($int64)
   type*                  string
                          Enum:

                              Array [ 3 ]
}
```

```
Team Member Dto    {
   description:
                          Required details needed to create a new Team Member

   dateOfBirth            string($date)
   firstName              string
   lastName               string
   type                   string
                          Enum:

                              Array [ 3 ]
}
```