Soccer Manager Server API Description

Database : MongoDB

1. 회원관리

1-1. 회원가입 API

URL : /member/register
method : POST
회원정보를 받아 DB에 기록한다.

Request

Key	Description	Example
id	아이디	
password	비밀번호	
name	이름	
birth	생년월일	900912
location	지역	경기도 연천
preferredPosition	선호포지션	공격수, 미드필더, 수비수, 골키퍼
myTeamName	소속팀 이름	전곡라이언즈 default : null
mySpeed	속력	1~5 default : 0
acceleration	가속력	1~5 default : 0
health	체력	1~5 default : 0
agility	민첩성	1~5 default : 0
captain	주장	true, false default : null

Response(JSON으로 응답)

```
{
  status : //상태( 성공(success), 실패(fail), 아이디중복(overlap) )
  reason : //실패시 사유, 성공(“회원가입 성공”)
  resultData : null
}
```

1-2. 회원정보 수정 API

URL : /member/modify/:id
method : POST
회원정보를 받아 DB의 내용을 수정 한다.

Request

Key	Description	Example
id	아이디	
password	비밀번호	
name	이름	
birth	생년월일	
location	지역	
preferredPosition	선호포지션	
myTeamName	소속팀 이름	
mySpeed	속력	
acceleration	가속력	
health	체력	
agility	민첩성	
captain	주장	

Response(JSON으로 응답)

```
{
  status : //상태(성공(success),실패(fail))
  reason : //실패시 사유, 성공(“정보수정 성공”)
  resultData : null
}
```

1-3. 회원 전체 조회 API [관리자용]

URL : /member/findAll
method : GET
DB 내의 모든 회원정보를 조회한다.

Request

Key	Description	Example
null	빈 값	조건없이 모두 검색

Response

```
{
   "status":  //상태(성공(success),실패(fail))
   "reason": //실패시 사유, 성공(“전체조회 성공”)
   "resultData": [
      {
         "id": "아이디",
         "password": "비밀번호",
         "name": "이름",
         "birth": "생년월일(900912)",
         "location": "회원 활동 지역(경기도 연천)",
         "preferredPosition": "선호포지션(f,m,c,g)",
         "myTeamName": "소속팀 이름",
         "mySpeed": "속력(1~5)",
         "acceleration": "가속력(1~5)",
         "health": "체력(1~5)",
         "agility": "민첩성(1~5)",
         "captain": "주장(true, false)"
      },
      {
                  ...
      },
      {
                  ...
         ]
}
```

1-4. 특정 회원 조회 API

URL : /member/findUser/:id
method : GET
특정 회원을 검색한다.

Request

Key	Description	Example
id	아이디	검색 아이디

Response(JSON으로 응답)

```
{
  status :  //상태( 성공(success), 실패(fail), 아이디 없음(not find) )
  reason : //실패시 사유, 성공(“유저 검색 성공”)
  resultData : {
    "id": "아이디",
    "name": "이름",
    "birth": "생년월일(900912)",
    "location": "회원 활동 지역(경기도 연천)",
    "preferredPosition": "선호포지션(f,m,c,g)",
    "myTeamName": "소속팀 이름",
    "mySpeed": "속력(1~5)",
    "acceleration": "가속력(1~5)",
    "health": "체력(1~5)",
    "agility": "민첩성(1~5)",
    "captain": "주장(true, false)"
  }
}
```

1-5. 로그인 API

URL : /member/login
method : POST

Request

Key	Description	Example
id	아이디	
password	비밀번호	

Response(JSON으로 응답)

```
{
  status : //상태( 성공(success), 실패(fail), 불일치(not find) )
  reason : //실패시 사유, 성공(“로그인 성공”)
  resultData : {
    "id": "아이디",
    “password” : “비밀번호”,
    "name": "이름",
    "birth": "생년월일(900912)",
    "location": "회원 활동 지역(경기도 연천)",
    "preferredPosition": "선호포지션(f,m,c,g)",
    "myTeamName": "소속팀 이름",
    "mySpeed": "속력(1~5)",
    "acceleration": "가속력(1~5)",
    "health": "체력(1~5)",
    "agility": "민첩성(1~5)",
    "captain": "주장(true, false)"
  }
}
```

1-6. 회원탈퇴 API

URL : /member/delete/:id
method : GET

Request

Key	Description	Example
id	아이디	

Response(JSON으로 응답)

```
{
  status : //상태(성공(success),실패(fail))
  reason : //실패시 사유, 성공(“회원탈퇴 성공”)
  resultData : {
    "n": 1,
    "ok": 1
  }
}
```

2. 구단관리

2-1. 구단생성 API

URL : /team/create
method : POST
구단정보를 받아 DB에 기록한다.

Request

Key	Description	Example
name	구단명	
captain	주장 이름	구단 생성한 유저
location	구단 활동지역	

Response(JSON으로 응답)

```
{
  status : //상태( 성공(success), 실패(fail), 구단명 중복(overlap) )
  reason : //실패시 사유, 성공(“구단생성 성공”)
  resultData : {
    "name": "구단명",
    "count": "단원 수", // 기본값 : 1
    "captain": "주장 이름",
    "location": "구단 활동지역"
  }
}
```

2-2. 구단가입 API

URL : /team/join
method : POST
회원정보와 구단정보를 갱신한다.

Request

Key	Description	Example
id	아이디	가입을 요청하는 아이디
name	구단 이름	가입하는 구단 이름

Response(JSON으로 응답)

```
{
  status : //상태( 성공(success), 실패(fail), 이미 다른 구단이 있음(already) )
  reason : //실패시 사유, 성공(“구단가입 성공”)
  resultData : {
    null
  }
}
```

2-3. 구단 전체 조회 API [관리자용]

URL : /team/teamlist
method : GET
DB 내의 모든 구단정보를 조회한다.

Request

Key	Description	Example
null	빈 값	조건없이 모두 검색

Response

```
{
   "status":  //상태(성공(success),실패(fail))
   "reason": //실패시 사유, 성공(“구단 조회 성공”)
   "resultData": [
      {
         "name": "구단 이름",
         "count": "단원 수",
         "captain": "주장 이름",
         "location": "구단 위치"
       },
      {
                  ...
      },
      {
                  ...
         ]
}
```

2-4. 특정 구단 조회 API

URL : /team/findTeam/:name
method : GET
name을 포함하는 모든 구단을 검색한다.

Request

Key	Description	Example
name	구단 이름	

Response(JSON으로 응답)

```
{
  status :  //상태( 성공(success), 실패(fail), 아이디 없음(not find) )
  reason : //실패시 사유, 성공(“유저 검색 성공”)
  resultData : [
    {
    "name": "구단명",
    "count": "단원 수",
    "captain": "주장 이름",
    "location": "구단 위치"
    },
    {
        ……      // name이 포함되는 모든 구단을 검색
    }
  ]
}
```

2-5. 구단명 / 구단 주장 변경 API

URL : /team/modify/:id
method : POST
회원정보와 구단정보를 갱신한다.

Request

Key	Description	Example
id	아이디	변경을 요청하는 아이디
name	구단 이름	변경할 구단 이름
captainName	주장 이름	변경할 주장 이름

Response(JSON으로 응답)

```
// 공통 ( 구단이 존재하고, 주장이 맞는지 검사 )
{
   status : //상태( 성공(success), 실패(fail), 가입 구단없음 (not find), 권한 없음 (auth false)
)
   reason : //실패시 사유, 성공(“권한 있음”)
   resultData : {
     "name": "구단 이름",
     "count": "단원 수",
     "captain": "주장 이름",
     "location": "구단 위치"
   }
}

// 구단명 변경
{
  {
   status : //상태( 성공(success), 실패(fail) )
   reason : //실패시 사유, 성공(“구단명 변경 성공”)
   resultData : {
     null
   }
}

or

// 주장 변경
{
  {
   status : //상태( 성공(success), 실패(fail) )
   reason : //실패시 사유, 성공(“주장 변경 성공”)
   resultData : {
     null
   }
}
```

2-9. 내 구단 검색 API

URL : /team/myteam/:id
method : GET
DB에서 내 구단 정보를 가져온다.

Request

Key	Description	Example
id	아이디	검색을 요청하는 아이디

Response(JSON으로 응답)

```
{
  status : //상태( 성공(success), 실패(fail), 구단 없음(not find) )
  reason : //실패시 사유, 성공(“구단가입 성공”)
  resultData : {
    "name": "구단명",
    "count": "단원 수",
    "captain": "주장 이름",
    "location": "구단 활동지역"
  }
}
```

2-7. 내 구단의 팀원정보 검색 API

URL : /team/myteam/search/:name
method : GET
DB에서 내 구단의 팀원정보를 가져온다.

Request

Key	Description	Example
name	구단 명	

Response(JSON으로 응답)

```
{
  status : //상태( 성공(success), 실패(fail) )
  reason : //실패시 사유, 성공(“내 팀원 조회  성공”)
  resultData : [
    {
      "id": "아이디",
      "name": "이름",
      "birth": "생년월일(900912)",
      "location": "회원 활동 지역(경기도 연천)",
      "preferredPosition": "선호포지션(f,m,c,g)",
      "myTeamName": "소속팀 이름",      // 검색하는 구단 이름
      "mySpeed": "속력(1~5)",
      "acceleration": "가속력(1~5)",
      "health": "체력(1~5)",
      "agility": "민첩성(1~5)",
      "captain": "주장(true, false)"
    },
    {
      …
    },
      ...
    ]
}
```

2-8. 구단 해체 API

URL : /team/delete/:id
method : GET
구단을 해체한다.
Request

Key	Description	Example
id	아이디	해체를 요청하는 아이디

Response(JSON으로 응답)

```
{
  status : //상태( 성공(success), 실패(fail), 권한 오류(auth false), 구단 없음(not find) )
  reason : //실패시 사유, 성공(“구단해체 성공”)
  resultData : {
    null
  }
}
```

2-9. 구단 탈퇴 API

URL : /team/quit
method : POST
구단을 해체한다.
Request

Key	Description	Example
id	아이디	탈퇴 요청하는 아이디
name	구단 명	탈퇴할 구단 명
captain	주장 여부	true , false

Response(JSON으로 응답)

```
{
  status : //상태( 성공(success), 실패(fail), 구단 양도 (captain), 구단 없음(not find) )
  reason : //실패시 사유, 성공(“구단탈퇴 성공”)
  resultData : {
    null
  }
}
```

3. 게시판 관리

3-1. 글쓰기 API

URL : /board/write
method : POST
글 정보를 받아 DB에 기록한다.

Request

Key	Description	Example
title	글 제목	
id	글쓴이 아이디	
writer	글쓴이 이름	
teamName	게시판 소유 구단 명	
contents	글 내용	

Response(JSON으로 응답)

```
{
  status : //상태( 성공(success), 실패(fail) )
  reason : //실패시 사유, 성공(“글쓰기 성공”)
  resultData : {
    "title": "글 제목",
    "id": "글쓴이 아이디",
    "writer": "글쓴이 이름",
    "teamName": "구단 명",
    "contents": "글 내용",
    "time": "글 작성시간"  //  'YYYY-MM-DD HH24:MI:SS'
  }
}
```

3-2. 글조회 API

URL : /board/search/:name
method : GET
게시판 글정보를 조회한다.

Request

Key	Description	Example
name	구단 명	

Response(JSON으로 응답)

```
{
  status : //상태( 성공(success), 실패(fail) )
  reason : //실패시 사유, 성공(“글쓰기 성공”)
  resultData : [
    {
      "title": "글 제목",
      "id": "글쓴이 아이디",
      "writer": "글쓴이 이름",
      "teamName": "구단 명",
      "contents": "글 내용",
      "time": "글 작성시간"
    },
    {
      ...
    },
      ...
  ]
}
```

3-3. 글삭제 API

URL : /board/delete/:userId
method : POST
글 정보를 삭제한다.

Request

Key	Description	Example
userId	접근 아이디	
id	글쓴이 아이디	
time	글 작성 시간	

Response(JSON으로 응답)

```
{
   status : //상태( 성공(success), 실패(fail) )
   reason : //실패시 사유, 성공(“글삭제 성공”)
   resultData : {
      "n": 0,
       "ok": 1
   }
}
```