API

까마귀공방

BASE_URL : https://까마귀공방.com/api

목차

Authorization

- jwt : 회원가입하거나 로그인시 발급

API 목록

회원관리

회원가입

회원가입 API

POST | BASE_URL + /users/signup

- RequestBody

```
{
  "userId": String,
  "userPassword": String,
  "userNickname": String
}
```

- Response

```
{
  "result" : String
  "jwt" : String (성공 시)
}
```

Code

- 200 : OK
- 400 : Bad Request (Request 조건에 맞지 않는 입력)
- 409 : Conflict (중복 아이디)

로그인

로그인 API

POST | BASE_URL + /users/login

- RequestBody

```
{
  "userId": String,
  "userPassword": String
}
```

- Response

```
{
  "result" : String
  "jwt" : String (성공 시)
}
```

Code

- 200 : OK
- 400 : Bad Request (Request 조건에 맞지 않는 입력)
- 409 : Conflict (틀린 비밀번호 혹은 없는 아이디인 경우)

회원 정보 조회

각 회원의 정보를 조회하는 API

GET | BASE_URL + /users/info

- RequestHeader

```
{ "Authorization" : <jwt> }
```

- Response

```
{
  "result" : String,
  "userSeq" : Long,
  "userId" : String,
  "userNickname" : String,
  "userProfile" : String,
  "userGitUsername" : String,
  "userGitToken" : String
}
```

Code

- 200 : OK
- 400 : Bad Request
- 401 : Unauthorized (JWT가 없거나 틀린 경우)
- 404 : Not found (JWT에 해당하는 사용자가 없는 경우)

닉네임 수정

유저 닉네임을 수정하는 API

> **PUT** | BASE_URL + /users/edit/nickname

- RequestHeader

```
{ "Authorization" : <jwt> }
```

- RequestBody

```
{ "userNickname": String }
```

- Response

```
{
  "result" : String,
  "userNickname": String (성공 시)
}
```

Code

- 200 : OK
- 400 : Bad Request
- 401 : Unauthorized (JWT가 없거나 틀린 경우)
- 404 : Not found (JWT에 해당하는 사용자가 없는 경우)

프로필 사진 수정

200 400 401 404

프로필 사진 삭제

200 400 401 404

Git 유저명, 토큰 수정

사용자의 Git유저명, 토큰 수정하는 API

> **PUT** | BASE_URL + /users/edit/git

- RequestHeader

```
{ "Authorization" : <jwt> }
```

- RequestBody

```
{
  "userGitUsername": String,
  "userGitToken" : String
}
```

- Response

```
{
  "result" : String,
  "userGitUsername" : String (성공 시)
}
```

Code

- 200 : OK
- 400 : Bad Request
- 401 : Unauthorized (JWT가 없거나 틀린 경우)
- 404 : Not found (JWT에 해당하는 사용자가 없는 경우)

비밀번호 수정

유저의 비밀번호를 수정하는 API

PUT | BASE_URL + /users/edit/password

- RequestHeader

```
{ "Authorization" : <jwt> }
```

- RequestBody

```
{
  "userPassword": String,
  "userNewPassword" : String
}
```

- Response

```
{ "result" : String }
```

Code

- 200 : OK
- 400 : Bad Request
- 401 : Unauthorized (JWT가 없거나 틀린 경우)
- 404 : Not found (JWT에 해당하는 사용자가 없는 경우)
- 409 : Conflict (비밀번호 틀린 경우)

회원 탈퇴

회원 탈퇴 API

DELETE | BASE_URL + /users/quit

- RequestHeader

```
{ "Authorization" : <jwt> }
```

- Response

```
{ "result" : String }
```

Code

- 200 : OK
- 400 : Bad Request
- 401 : Unauthorized (JWT가 없거나 틀린 경우)
- 403 : Forbidden (팀장인 유저가 탈퇴하려는 경우)
- 404 : Not found (JWT에 해당하는 사용자가 없는 경우)

개인 환경 세팅 저장

사용자별 개인 환경 세팅을 저장하는 API

PUT | BASE_URL + /users/personal/{teamSeq}

- PathVariable
  - teamSeq : 저장하고자 하는 프로젝트의 팀 Sequence
- RequestHeader

```
{ "Authorization" : <jwt> }
```

- RequestBody

```
{
      "result" : String,
      "horizonSplit": int
      "lastTab": [
          {
              "name": String,
              "path": String,
              "isLast" : String // "Y" or "N"
          },
          ...
      ],
      "lastSideBar": String,
```

```
      "editors": {
          "fontSize": int,
          "font": String,
          "autoLine" : String // "on" or "off"
      },
      "consoles": {
          "fontSize": int,
          "font": String
      }
}
```

- Response

```
{ "result" : String }
```

Code

- 200 : OK
- 400 : Bad Request
- 401 : Unauthorized (JWT가 없거나 틀린 경우)
- 403 : Forbidden (해당 팀에 속해있지 않을 경우)
- 404 : Not found (JWT에 해당하는 사용자가 없는 경우)

개인 환경 세팅 조회

사용자별 개인 환경 세팅을 조회하는 API

GET | BASE_URL + /users/personal/{teamSeq}

- PathVariable
  - teamSeq : 저장하고자 하는 프로젝트의 팀 Sequence
- RequestHeader

```
{ "Authorization" : <jwt> }
```

- Response

```
{
      "result" : String,
      "horizonSplit": int
      "lastTab": [
          {
              "name": String,
              "path": String,
              "isLast" : String // "Y" or "N"
          },
          ...
      ],
      "lastSideBar": String,
      "editors": {
          "fontSize": int,
          "font": String,
          "autoLine" : String // "on" or "off"
      },
      "consoles": {
          "fontSize": int,
          "font": String
      }
}
```

Code

- 200 : OK
- 400 : Bad Request
- 401 : Unauthorized (JWT가 없거나 틀린 경우)
- 403 : Forbidden (해당 팀에 속해있지 않을 경우)
- 404 : Not found (JWT에 해당하는 사용자가 없는 경우)

프로필 조회

사용자의 프로필을 유저 Sequence로 조회하는 API

> **GET** | BASE_URL + /users/mypage/{userSeq}

- PathVariable
  - userSeq : 프로필을 조회할 유저의 Sequence
- RequestHeader

```
{ "Authorization" : <jwt> }
```

- Response

```
{
  "result" : String,
  "userSeq" : Long,
  "userId" : String,
  "userNickname" : String,
  "userProfile" : String
}
```

Code

- 200 : OK
- 400 : Bad Request
- 404 : Not Found

유저 검색

유저를 검색하는 API

> **GET** | BASE_URL + /users/search

- RequestBody

```
{ "searchWord" : String }
```

- Response
  - 없으면 빈 리스트

```
[
    {
        "result": String,
        "userSeq": Long,
        "userId": String,
        "userNickname": String,
        "userProfile": String
    },
  ...
]
```

Code

- 200 : OK
- 400 : Bad Request

컴파일러

Python 컴파일러

Pure Python 파일이나 Django, Flask, FastAPI 프로젝트를 컴파일하는 API

Docker를 활용해 컴파일하고 프로젝트일 경우 서버를 띄워준다

POST | BASE_URL + /compile/py

- RequestHeader

```
{ "Authorization" : <jwt> }
```

- RequestBody
  - type : 컴파일하려는 파일(프로젝트)의 종류
    1. Pure Python File
    2. Django project
    3. Flask project
    4. FastAPI project
  - filePath : 컴파일할 프로젝트 경로 혹은 파이썬 파일 경로 (`teamSeq/teamName` or `teamSeq/.../example.py`)
  - input : input할 값이 있을 경우 input값, 없는 경우 빈 문자열

```
{
  "type" : String,
  "filePath" : String,
  "input" : String
}
```

- Response

```
{
  "result" : String,  // SUCCESS ..
  "response" : String // 실제 결과
  "message" : String  // 오류메세지 (없으면 빈문자열)
 }
# print문이면 hello world 이렇게 가고 플젝이면 k7d207.. 으로 서버 주소만 갑니당
```

  Code

  - 200 : OK
  - 400 : Bad Request (컴파일을 할 수 없는 경우)
  - 401 : Unauthorized (JWT가 없거나 틀린 경우)

컴파일러 중단

실행되고 있는 프로젝트를 중단하는 API

Docker Container를 중단 & 삭제, Docker image를 삭제한다

(pure Python파일은 X)

POST | BASE_URL + /compile/py/stop

- RequestHeader

```
{ "Authorization" : <jwt> }
```

- RequestBody

```
{
  "teamName" : String,
  "teamSeq" : String
}
```

- Response

```
{ "result" : String }
```

  Code

- 200 : OK
- 400 : Bad Request
- 401 : Unauthorized (JWT가 없거나 틀린 경우)

API 응답 테스트

api 응답 테스트

프로젝트를 실행한 후 본인의 API의 응답 시간을 테스트하는 API

이 API로 사용자가 작성한 API가 잘 동작하는지 확인 가능하다

> **POST** | BASE_URL + /api/api-test

- RequestHeader

```
{ "Authorization" : <jwt> }
```

- RequesetBody
  - api : 응답테스트를 진행할 API URI
  - type : API의 요청방식 (get, post, put, delete)
  - request : 해당 API의 request
  - header : 해당 API의 header

```
{
  "api" : String,
  "type" : String,
  "request" : JSON,
  "header": JSON
}
```

- Response

```
{
  "result" : String,
  "data": JSON,
```

```
  "time": Int
}
```

Code

- 200 : OK
- 400 : Bad Request
- 401 : Unauthorized (JWT가 없거나 틀린 경우)

코드 에디터

린트(lint)

사용자 코드의 오류를 잡는 린트 API

pylint를 활용하여 구현

POST | BASE_URL + /editors/lint/{language}

- PathVariable
  - language : 린트할 파일의 프로그래밍 언어 종류 ex) python

    (현재 python만 가능)
- RequestHeader

```
{ "Authorization" : <jwt> }
```

- RequestBody
  - text : 린트를 실행할 코드

```
{ "text" : String }
```

- Response

```
{
  "status" : Int,
  "data" : List<String>,
```

```
  "index" : List<Int>
}
```

Code

- 200 : OK
- 400 : Bad Request

- Example

```
{
    "status": 200,
    "data": [
        "Missing module docstring (missing-module-docstring)",
        "Missing function or method docstring (missing-function-docstring)",
        "Constant name \"a\" doesn't conform to UPPER_CASE naming style (invalid-name)",
        "Undefined variable 'b' (undefined-variable)"
    ],
    "index": [
        1,
        6,
        10,
        12
    ]
}
```


포맷팅(Formatting)

코드를 다듬어주는 formatting API

black을 활용하여 구현

> **POST** | BASE_URL + /editors/format/{language}

- PathVariable
  - language : 포맷팅할 파일의 프로그래밍 언어 종류 ex) python, text
- RequestHeader

```
{ "Authorization" : <jwt> }
```

- RequestBody
  - text : 포맷팅할 문자열

```
{ "text" : String }
```

- Response

```
{
  "result" : String,
  "data" : String (result가 SUCCESS일 때만)
}
```

  Code
  - 200 : OK
  - 400 : Bad Request (포맷팅 실패한 경우)
  - 401 : Unauthorized (JWT가 없거나 틀린 경우)

포맷팅 결과 조회

포맷팅 후 결과 조회하는 API

POST | BASE_URL + /format/read/{language}

- PathVariable
  - language : 조회할 파일의 프로그래밍 언어 종류 ex) python
- RequestHeader

```
{ "Authorization" : <jwt> }
```

- RequestBody
  - name : 포맷팅에서 받은 data의 값(String)

```
{ "name": String }
```

- Response

```
{
  "result" : String,
  "data" : String   (result가 SUCCESS일 때만)
}
```

Code

- 200 : OK
- 400 : Bad Request
- 401 : Unauthorized (JWT가 없거나 틀린 경우)

변수명 추천

변수명 추천

변수명을 추천받는 API

변수명을 추천받고싶은 한국어를 입력하면 변수명을 추천해준다

Papago API를 활용하여 구현

> **POST** | BASE_URL + /variable

- RequestHeader

```
{ "Authorization" : <jwt> }
```

- RequestBody
  - data : 변수명을 추천받을 단어

```
{ "data": String }
```

- Response

```
{
  "result" : String,
  "data": List<String>
}
```

Code

- 200 : OK
- 400 : Bad Request
- 401 : Unauthorized (JWT가 없거나 틀린 경우)

- Example
  - RequestBody

    ```
    { "data": "객체탐지" }
    ```

  - Response

    ```
    {
        "data": [
            "objectDetection",
            "ObjectDetection",
            "object_detection"
        ]
    }
    ```


팀

사용자가 속한 팀 목록을 조회하는 API

GET | BASE_URL + /teams

- RequestHeader

```
{ "Authorization" : <jwt> }
```

- Response

```
[
  {
    "result" : String,
    "teamSeq" : Long,
    "teamName" : String,
    "teamLeaderSeq" : Long,
    "teamLeaderNickname" : String,
    "teamLeaderProfile" : String,
    "teamGit" : String,
    "projectType" : String,
    "teamMember" : [
      {
        "memberSeq" : Long,
        "memberNickname" : String,
        "memberProfile" : String
      },
      ...
    ]
  },
  ...
]
```

Code

- 200 : OK
- 400 : Bad Request
- 401 : Unauthorized (JWT가 없거나 틀린 경우)
- 404 : Not Found

팀 조회

팀의 세부 정보를 조회하는 API

GET | BASE_URL + /teams/{teamSeq}

- PathVariable
  - teamSeq : 조회할 팀의 Sequence
- RequestHeader

```
{ "Authorization" : <jwt> }
```

- Response

```
{
  "result" : String,
  "teamSeq" : Long,
  "teamName" : String,
  "teamLeaderSeq" : Long,
  "teamLeaderNickname" : String,
  "teamLeaderProfile" : String,
  "teamGit" : String,
  "projectType" : String,
  "teamMember" :
    [
      {
        "memberSeq" : Long,
        "memberNickname" : String,
        "memberProfile" : String
      },
      ...
    ]
}
```

Code

- 200 : OK
- 400 : Bad Request
- 401 : Unauthorized (JWT가 없거나 틀린 경우)
- 403 : Forbidden (조회할 팀에 해당 유저가 멤버가 아닌 경우)
- 404 : Not found (조회하는 팀이 없는 경우)

팀 생성

팀을 생성하는 API

- 팀장 자동 설정

POST | BASE_URL + /teams/create

- RequestHeader

```
{ "Authorization" : <jwt> }
```

- RequestBody
  - projectType : 생성하는 팀 생성
    1. Pure Python File
    2. Django
    3. Flask
    4. FastAPI
  - projectGit
    - git clone하는 경우 : Repository URL
    - git clone하지 않는 경우 : null

```
{
  "teamName" : String,
  "projectType" : String,
  "projectGit" : String
}
```

- Response

```
{
  "result" : String,
  "teamSeq" : String (성공 시)
}
```

Code

- 200 : OK
- 400 : Bad Request
- 401 : Unauthorized (JWT가 없거나 틀린 경우)
- 404 : Not Found (Git Repository URL을 찾을 수 없는 경우)
- 409 : Conflict (리더와 팀명이 모두 같은 팀이 이미 있는 경우)

팀명 수정

팀명을 수정하는 API

- 팀장 권한

PUT | BASE_URL + /teams/modify/name/{teamSeq}

- PathVariable
  - teamSeq : 수정할 팀의 Sequence
- RequestHeader

```
{ "Authorization" : <jwt> }
```

- RequestBody

```
{ "teamName" : String }
```

- Response ← key-value

```
{
  "result" : String,
  "teamName" : String (성공 시)
}
```

  - 200 : OK
  - 400 : Bad Request
  - 401 : Unauthorized (JWT가 없거나 틀린 경우)
  - 403 : Forbidden
  - 404 : Not Found
  - 409 : Conflict (팀장과 팀명이 모두 같은 팀이 존재하여 변경이 불가능한 경우)

팀 Git 수정

팀의 Git을 수정하는 API

- 팀장 권한

PUT | BASE_URL + /teams/modify/git/{teamSeq}

- PathVariable
  - teamSeq : 수정할 팀의 Sequence
- RequestHeader

```
{ "Authorization" : <jwt> }
```

- RequestBody

```
{ "teamGit" : String }
```

- Response ← key-value

```
{
  "result" : String,
  "teamGit" : String (성공 시)
}
```

  Code
  - 200 : OK
  - 400 : Bad Request
  - 401 : Unauthorized (JWT가 없거나 틀린 경우)
  - 403 : Forbidden
  - 404 : Not Found

팀 프로젝트 타입 수정

팀의 프로젝트 타입을 수정하는 API

- 팀장 권한

> **PUT** | BASE_URL + /modify/type/{teamSeq}

- PathVariable
  - teamSeq : 수정할 팀의 Sequence

- RequestHeader

```
{ "Authorization" : <jwt> }
```

- RequestBody
  - projectType

```
{ "projecetType" : String }
```

- Response ← key-value

```
{
  "result" : String,
  "projectType" : String (성공 시)
}
```

  Code
  - 200 : OK
  - 400 : Bad Request
  - 401 : Unauthorized (JWT가 없거나 틀린 경우)
  - 403 : Forbidden
  - 404 : Not Found

팀 삭제

팀을 삭제하는 API

- 팀장 권한

> **DELETE** | BASE_URL + /teams/delete/{teamSeq}

- PathVariable
  - teamSeq : 삭제할 팀의 Sequence
- RequestHeader

```
{ "Authorization" : <jwt> }
```

- Response

```
{ "result" : String }
```

Code

- 200 : OK
- 400 : Bad Request
- 401 : Unauthorized (JWT가 없거나 틀린 경우)
- 403 : Forbidden
- 404 : Not Found

팀원 목록 조회

팀의 팀원 목록을 조회하는 API

GET | BASE_URL + /teams/member/{teamSeq}

- PathVariable
  - teamSeq : 조회할 팀의 Sequence
- RequestHeader

```
{ "Authorization" : <jwt> }
```

- Response

```
[
  {
    "userSeq" : Long,
    "userId" : String,
    "userNickname" : String,
    "userProfile" : String
  },
  ...
]
```

Code

- 200 : OK
- 400 : Bad Request
- 401 : Unauthorized (JWT가 없거나 틀린 경우)
- 403 : Forbidden (해당 유저가 멤버가 아닌 팀이기 때문에 조회가 불가능한 경우)
- 404 : Not Found (조회하는 팀이 존재하지 않는 경우)

팀원 추가

팀에 팀원을 추가하는 API

- 팀장 권한

PUT | BASE_URL + /teams/add

- RequestHeader

```
{ "Authorization" : <jwt> }
```

- RequestBody

```
{
  "teamSeq" : Long,
  "memberSeq" : Long
}
```

- Response

```
{ "result" : String }
```

Code

- 200 : OK
- 400 : Bad Request

- 401 : Unauthorized (JWT가 없거나 틀린 경우)
- 403 : Forbidden (해당 유저가 팀장이 아니기 때문에 권한이 없는 경우)
- 404 : Not Found (추가하려는 팀이 존재하지 않는 경우)
- 409 : Conflict (이미 그 팀의 멤버일 경우)

팀원 삭제

팀원을 삭제하는 API

- 팀장 권한

> **DELETE** | BASE_URL + /teams/remove

- RequestHeader

```
{ "Authorization" : <jwt> }
```

- RequestBody

```
{
  "teamSeq" : Long,
  "memberSeq" : Long
}
```

- Response

```
{ "result" : String }
```

  Code

  - 200 : OK
  - 400 : Bad Request
  - 401 : Unauthorized (JWT가 없거나 틀린 경우)
  - 403 : Forbidden (팀장이 아니기 때문에 권한이 없는 경우)
  - 404 : Not Found (팀이 존재하지 않는 경우)

- 409 : Conflict (멤버가 아닌 유저를 탈퇴시키거나 스스로를 탈퇴시키려 한 경우)

팀장 위임

팀의 팀장을 위임하는 API

- 팀장 권한

PUT | BASE_URL + /teams/beLeader

- RequestHeader

```
{ "Authorization" : <jwt> }
```

- RequestBody
  - memberSeq : 위임받을 유저의 Sequence

```
{
  "teamSeq" : Long,
  "memberSeq" : Long
}
```

- Response

```
{ "result" : String }
```

Code

- 200 : OK
- 400 : Bad Request
- 401 : Unauthorized (JWT가 없거나 틀린 경우)
- 403 : Forbidden (팀장이 아니기 때문에 권한이 없는 경우)
- 404 : Not Found (팀이 존재하지 않는 경우)
- 409 : Conflict (멤버가 아닌 유저를 탈퇴시키거나 스스로를 탈퇴시키려 한 경우)

팀 탈퇴

팀을 탈퇴하는 API

- 팀장은 탈퇴 불가

DELETE | BASE_URL + /quit/{teamSeq}

- PathVariable
  - teamSeq : 탈퇴할 팀의 Sequence
- RequestHeader

```
{ "Authorization" : <jwt> }
```

- Response

```
{ "result" : String }
```

  Code
  - 200 : OK
  - 400 : Bad Request
  - 401 : Unauthorized (JWT가 없거나 틀린 경우)
  - 403 : Forbidden (팀장이기 때문에 팀 탈퇴 권한이 없는 경우)
  - 404 : Not Found (팀이 존재하지 않는 경우)
  - 409 : Conflict (해당 유저가 해당 팀의 멤버가 아닌 경우)

프로젝트

프로젝트 디렉토리 조회

프로젝트의 파일 구조를 조회하는 API

POST | BASE_URL + /projects/directories/{teamSeq}

- PathVariable
  - teamSeq : 조회하려는 프로젝트(팀)의 Sequence
- RequestHeader

```
{ "Authorization" : <jwt> }
```

- Response

```
{
  "children" :
    [
      {
        "children" : [
          ],
        "name" : String,
        "id" : String
      }
    ],
  "name" : String,
  "id" : String,
  "type" : String,
}
```

type

- python
- html
- js
- css
- text
- folder

Code

- 200 : OK
- 400 : Bad Request

- 401 : Unauthorized (JWT가 없거나 틀린 경우)

프로젝트 생성

프로젝트를 생성하는 API

- 프로젝트별 탬플릿 제공

> **POST |** BASE_URL + /projects/{teamSeq}?type={type번호}

- PathVariable
  - teamSeq : 프로젝트를 생성할 팀의 Sequence
- Parameter : 생성할 프로젝트의 종류
  - type
    1. Pure Python File
    2. Django
    3. Flask
    4. FastAPI
- RequestHeader

```
{ "Authorization" : <jwt> }
```

- RequestBody

```
{ "projectName" : Int }
```

- Response

```
{ "result" : String }
```

  Code
  - 200 : OK
  - 400 : Bad Request

- 401 : Unauthorized (JWT가 없거나 틀린 경우)

파일

파일생성

새 파일 혹은 폴더를 생성하는 API

> **POST** | BASE_URL + /files/{teamSeq}?type={type번호}

- PathVariable
  - teamSeq : 파일을 생성할 팀의 Sequence
- Parameter : 생성할 문서의 종류
  - type
    1. Directory (폴더)
    2. File (파일)
- RequestHeader

```
{ "Authorization" : <jwt> }
```

- RequestBody

```
{
  "fileTitle" : String,
  "filePath" : String
}
```

- Response

```
{
 "result" : String
 "filePath" : String
}
```

Code

- 200 : OK
- 400 : Bad Request
- 401 : Unauthorized (JWT가 없거나 틀린 경우)

파일 삭제

파일 혹은 폴더를 삭제하는 API

> **DELETE** | BASE_URL + /files/{teamSeq}?type={type번호}

- PathVariable
  - teamSeq : 파일을 삭제할 팀의 Sequence
- Parameter : 삭제할 문서의 종류
  - type
    1. Directory (폴더)
    2. File (파일)
- RequestHeader

```
{ "Authorization" : <jwt> }
```

- RequestBody

```
{ "filePath" : String }
ex : {"filePath" : "82/ole/rerendering"}
```

- Response

```
{ "result" : String }
```

Code

- 200 : OK

- 400 : Bad Request
- 401 : Unauthorized (JWT가 없거나 틀린 경우)

파일 저장

파일을 저장하는 API

> **PUT** | BASE_URL + /files/{teamSeq}/files

- PathVariable
  - teamSeq : 파일을 저장할 팀의 Sequence
- RequestHeader

```
{ "Authorization" : <jwt> }
```

- RequestBody

```
{
  "filePath" : String,
  "fileContent" : String,
}
```

- Response

```
{ "result" : String }
```

  Code
  - 200 : OK
  - 400 : Bad Request
  - 401 : Unauthorized (JWT가 없거나 틀린 경우)

파일명 수정

파일명을 수정하는 API

PUT | BASE_URL + /files/{teamSeq}/file-title

- RequestHeader

```
{ "Authorization" : <jwt> }
```

- RequestBody
  - fileTitle : 변경할 파일명

```
{
  "filePath" : String,
  "oldFileName" : String,
  "fileTitle" : String
}
```

- Response

```
{ "result" : String }
```

Code

- 200 : OK
- 400 : Bad Request
- 401 : Unauthorized (JWT가 없거나 틀린 경우)

파일 조회

파일내용을 조회하는 API

POST | BASE_URL + /files/files

- RequestHeader

```
{ "Authorization" : <jwt> }
```

- RequestBody

```
{ "filePath" : String }
```

- Response

```
{
  "result" : String,
  "fileContent" : String (성공시)
}
```

  Code

  - 200 : OK
  - 400 : Bad Request
  - 401 : Unauthorized (JWT가 없거나 틀린 경우)

Git

Git Clone

Git에서 Clone받는 API

> **POST** | BASE_URL + /git/{teamSeq}

- PathVariable
  - teamSeq : 클론받을 팀의 Sequence
- RequestHeader

```
{ "Authorization" : <jwt> }
```

- RequestBody

```
{
  "projectName" : String,
  "gitUrl" : String
}
```

- Response

```
{ "result" : String }
```

  Code

  - 200 : OK
  - 400 : Bad Request
  - 401 : Unauthorized (JWT가 없거나 틀린 경우)

Git Switch

branch switch하는 API

> **POST** | BASE_URL + /git/{teamSeq}/git-switch?type={type 번호}

- Parameter
  - type : switch할 브랜치의 종류
    1. 존재하는 브랜치로 이동
    2. 브랜치를 새로 생성 후 이동
- RequestHeader

```
{ "Authorization" : <jwt> }
```

- RequestBody

```
{
  "branchName" : String
}
```

- Response

```
{ "result" : String }
```

  Code

- 200 : OK
- 400 : Bad Request
- 401 : Unauthorized (JWT가 없거나 틀린 경우)

Git branch 조회

Git branch 목록을 조회하는 API

> Get | BASE_URL + /git/{teamSeq}/branches?type={type 번호}

- Parameter
  - type : 조회하려는 브랜치의 종류
    1. local branch
    2. remote branch
- RequestHeader

```
{ "Authorization" : <jwt> }
```

- Response

```
List<String>
```

  Code
  - 200 : OK
  - 400 : Bad Request
  - 401 : Unauthorized (JWT가 없거나 틀린 경우)

Git Commit

Git commit하는 API

POST | BASE_URL + /git/{teamSeq}/git-commit

- RequestHeader

```
{ "Authorization" : <jwt> }
```

- RequestBody
  - message : 커밋 메세지
  - filePath : add할 파일 경로 (특정하지 않을 경우 all)

```
{
  "message" : String,
  "filePath" : String
}
```

- Response

```
{
  "result" : String,
  "message" : String
}
```

  Code
  - 200: OK
  - 401 : Unauthorized (JWT가 없거나 틀린 경우)

Git Push

Git push하는 API

POST | BASE_URL + /git/{userSeq}/git-push

- PathVariable
  - userSeq : push하는 유저의 Sequence
- RequestHeader

```
{ "Authorization" : <jwt> }
```

- RequestBody
  - message : commit 메세지
  - teamSeq : push할 팀 Sequence
  - filePath : add할 파일 경로 (특정하지 않을 경우 all)
  - branchName : push할 브랜치명

```
{
  "message" : String,
  "teamSeq" : String,
  "filePath" : String,
  "branchName" : String
}
```

- Response

```
{ "result" : String }
```

Code

- 200 : OK
- 401 : Unauthorized (JWT가 없거나 틀린 경우)

Git Pull

Git pull하는 API

> **POST** | BASE_URL + /git/{userSeq}/git-pull

- PathVariable
  - userSeq : pull할 유저의 Sequence
- RequestHeader

```
{ "Authorization" : <jwt> }
```

- RequestBody
  - teamSeq : push할 팀 Sequence
  - branchName : pull할 브랜치명

```
{
  "teamSeq" : String,
  "branchName" : String
}
```

- Response

```
{ "result" : String }
```

  Code
  - 200 : OK
  - 400 : Bad Request
  - 401 : Unauthorized (JWT가 없거나 틀린 경우)