Base url: <https://5ni1uif3uc.execute-api.ap-northeast-2.amazonaws.com/test/>

1. 로그인: email, pswd -> token

Path: user/login

Method: Post

Content Type: application/json

Request Body:

{

“email”: \_\_\_(String),

“pswd”: \_\_\_\_(String)

}

Response Body:

{

“status”: 0 or 1 (Integer)

“token”: \_\_\_(String),

“msg”: \_\_\_\_\_(String)

}

1. 비번찾기 이메일 인증코드 전송: email -> success or fail msg

Content Type: application/json

Path: user/password/sendmail/{email}

Method: PUT

Request Body:

{

“email”: \_\_\_\_\_\_(String)  
}

Response Body:

{

“status”:\_\_\_\_\_(Integer)

“msg”: \_\_\_\_\_\_\_(String)  
}

Status -> 0 이면 성공

-> 1 이면 해당 이메일은 가입되지 않음

-> 2 이면 해당 이메일에 보내기 실패

1. 비번찾기 이메일 인증코드 확인: email, code -> success or fail msg

Content Type: application/json

Path: user/password/verifycode

Method: PUT

Request Body:

{

“email”: \_\_\_\_\_ (String),

“code”: \_\_\_\_\_\_ (String)

}

Response Body:

{

“status”: \_\_\_\_\_(Integer)

“msg”: \_\_\_\_\_(String)

}

Status -> 0이면 성공

-> 1 이면 해당 이메일에 보내지 않았음

-> 2 이면 코드가 틀림

1. 비번찾기 새로운 비번 보내기: email, code, password - > success or fail msg

Content Type: application/json

Path: user/password

Method: PUT

Request Body:

{

“email”: \_\_\_\_ (String),

“code” : \_\_\_\_\_\_(String),

“password”: \_\_\_\_\_(String)

}

Response Body:

{

“status”: \_\_\_\_\_(Integer),

“msg”: \_\_\_\_\_\_\_(String)

}

Status -> 0이면 성공

-> 1 이면 해당 이메일이 인증되지 않음

-> 2 코드가 이메일에 보내지지 않음

-> 3 이면 코드가 틀림

1. 회원가입 이메일 인증코드 전송: email -> success or fail msg

Content Type: application/json

Path:user/email/sendcode/{email}

Method: POST

Response Body:

{

“status”:\_\_\_\_\_(Integer)

“msg”: \_\_\_\_\_\_\_(String)  
}

Status -> 0 이면 성공

-> 1 이면 해당 이메일에 보내기 실패

1. 회원가입 이메일 인증코드 확인: email, code -> success or fail msg

Content Type: application/json

Path: user/email/verifyCode

Method: POST

Request Body:

{

“email”: \_\_\_\_\_ (String),

“code”: \_\_\_\_\_\_ (String)

}

Response Body:

{

“status”: \_\_\_\_\_(Integer)

“msg”: \_\_\_\_\_(String)

}

Status -> 0이면 성공

-> 1 이면 해당 이메일에 보내지 않았음

-> 2 이면 코드가 틀림

1. 닉네임 중복확인: nickname -> success or fail msg

Content Type: application/json

Path: user/nickname/checkDuplicate/{nickname}

Method: GET

Response Body:

{

“status”: \_\_\_\_\_(Integer)

“msg”: \_\_\_\_\_(String)

}

Status -> 0이면 중복없음

-> 1 이면 중복있음

1. 회원가입: 이름, 이메일, 인증코드, 비밀번호, 닉네임 -> succcess or fail msg

Content Type: application/json

Path: user

Method: POST

Request Body:

{

“name”: \_\_\_\_\_\_(String),

“email”: \_\_\_\_\_ (String),

“code”: \_\_\_\_\_\_ (String),

“pswd”: \_\_\_\_\_\_(String),

“nickname”: \_\_\_\_(String)

}

Response Body:

{

“status”: \_\_\_\_\_(Integer)

“msg”: \_\_\_\_\_(String)

}

Status -> 0이면 성공

-> 1 이면 회원가입 실패

1. **기사추천: token -> [ {기사 제목, 사진url, 기사ID, path}]**

**쓰여진 날짜순으로 정렬해서 보낸다.**

Content Type: application/json

Path: articles/recommendation

Method: GET

Request Header:

{

“Authorization”: \_\_\_\_\_(String)

}

Response Body:

{

“status”: \_\_\_\_\_(Integer),

“msg”: \_\_\_\_\_(String),

“articles”: [

{

“pictUrl”:\_\_\_\_\_(String),

“path”: \_\_\_\_\_\_\_(String),

“title”: \_\_\_\_\_\_\_(String),

“fetchTime”:\_\_\_\_\_(Number)

},

{

“pictUrl”:\_\_\_\_\_(String),

“path”: \_\_\_\_\_\_\_(String),

“title”: \_\_\_\_\_\_\_(String),

“fetchTime”:\_\_\_\_\_(Number)

},

]

}

Status -> 0이면 성공

-> 1 이면 추천 실패

기사들은 생성 순서대로!

1. **기사 검색: keyword -> [{기사 제목, 사진URL, 기사ID, path}]**

**쓰여진 날짜순으로 정렬해서 보낸다.**

Content Type: application/json

Path: articles/search/{keyword}

Method:GET

Request Header:

{

“Authorization”: \_\_\_\_\_(String)

}

Response Body:

{

“status”: \_\_\_\_\_(Integer),

“msg”: \_\_\_\_\_(String),

“articles”: [

{

“pictUrl”:\_\_\_\_\_(String),

“path”: \_\_\_\_\_\_\_(String),

“title”: \_\_\_\_\_\_\_(String),

“fetchTime”: \_\_\_\_(Integer)

},

{

“pictUrl”:\_\_\_\_\_(String),

“path”: \_\_\_\_\_\_\_(String),

“title”: \_\_\_\_\_\_\_(String),

“fetchTime”: \_\_\_\_(Integer)

},

]

}

Status -> 0이면 성공

-> 1 이면 추천 실패

기사들은 생성 순서대로!

1. **방문기 보여줌: token -> [{방문기 ID, 방문기 제목, 주소, tags, 방문기에 좋아요를 눌렀는지, 글쓴이 닉네임, 조회수 }]**

Content Type: application/json

Path: visits/recommendation

Method: GET

Request Header:

{

“Authorization”: \_\_\_\_\_(String)

}

Response Body:

{

“status”: \_\_\_\_\_(Integer),

“msg”: \_\_\_\_\_(String),

“visits”: [

{

“title”: \_\_\_\_\_\_\_ (String),

“writer”: \_\_\_\_\_\_(String),

“heart”: \_\_\_\_\_ (Integer),

“address”: \_\_\_\_\_(String),

“tags”: [“”, “”]

},

{

“title”: \_\_\_\_\_\_\_ (String),

“writer”: \_\_\_\_\_\_(String),

“heart”: \_\_\_\_\_ (Integer),

“address”: \_\_\_\_\_(String),

“tags”: [“”, “”]

}

]

}

Status -> 0이면 성공

-> 1 이면 추천 실패

**쓰여진 날짜순으로 정렬해서 보낸다.**

1. **방문기 검색: token, 키워드(태그) ->[{방문기 ID, 방문기 제목, 주소, tags, 방문기에 좋아요를 눌렀는지 }]**

**Path: visits/search/{keyword}**

**Method: GET**

Content Type: application/json

Request Header:

{

“Authorization”: \_\_\_\_\_(String)

}

Response Body:

{

“status”: \_\_\_\_\_(Integer),

“msg”: \_\_\_\_\_(String),

“visits”: [

{

“title”: \_\_\_\_\_\_\_ (String),

“writer”: \_\_\_\_\_\_(String),

“heart”: \_\_\_\_\_ (Integer),

“address”: \_\_\_\_\_(String),

“tags”: [“”, “”]

},

{

“title”: \_\_\_\_\_\_\_ (String),

“writer”: \_\_\_\_\_\_(String),

“heart”: \_\_\_\_\_ (Integer),

“address”: \_\_\_\_\_(String),

“tags”: [“”, “”]

}

]

}

Status -> 0이면 성공

-> 1 이면 검색 실패

**쓰여진 날짜순으로 정렬해서 보낸다**.

1. **방문기 상세조회: 방문기ID -> 방문기ID, 방문기 제목, 주소, tags, 방문기에 좋아요를 눌렀는지, 조회수, 내용**

Content Type: application/json

Path: visits/writer/{writer}/title/{title}

Request Header:

{

“Authorization”: \_\_\_\_\_(String)

}

Response Body:

{

“status”: \_\_\_\_\_(Integer),

“msg”: \_\_\_\_\_(String),

“visit”:

{

“title”: \_\_\_\_\_\_\_ (String),

“writer”: \_\_\_\_\_\_(String),

“heart”: \_\_\_\_\_ (Integer),

“address”: \_\_\_\_\_(String),

“content”: [

{

"type": "text",

"content": "...."

},

{

"type":"spacer",

},

{

"type": "image",

"content": "<image\_url>"

}

],

“tags”: [“”, “”],

“nickname”: \_\_\_(String),

“readView”: \_\_\_\_(Integer),

“detailAddress”: \_\_\_\_\_(String)

}

}

Status -> 0이면 성공

-> 1 이면 추천 실패

1. 방문기 추가: token, 제목, 주소, 상세주소, 부동산타입, 태그들, content -> success or fail msg -v

Content Type: multipart/form-data

Path: visits

Method:POST

Request Header:

{  
 “Authorization”: \_\_\_\_\_(String)

}

Request Body:

{

“title”: \_\_\_\_\_\_\_ (String),

“address”: \_\_\_\_\_(String),

“content”: `[

{

"type": "text",

"content": "...."

},

{

"type":"spacer",

},

{

"type": "image",

}

]`,

“tags”: `[“”, “”]`,

“detailAddress”: \_\_\_\_\_(String)

“pict1”: \_\_\_(binary),

“pict2”: \_\_\_\_(binary)

}

* Pictures should be sent in order!

Response Body:

{

“msg”: \_\_\_\_(String),

“status”: \_\_\_\_(Integer),

visit: {

“title”: \_\_\_\_\_\_\_ (String),

“address”: \_\_\_\_\_(String),

“content”: [

{

"type": "text",

"content": "...."

},

{

"type":"spacer",

},

{

"type": "image",

"content": "<image\_url>"

}

],

“tags”: [“”, “”],

“detailAddress”: \_\_\_\_\_(String),

“readView”: \_\_\_\_,

“nickname” : \_\_\_\_(String)

}

}

1. 최근본 기사 조회: token -> [{사진URL, 기사 제목, 기사 ID, path}]

본날짜 순으로 정렬

Content Type: application/json

Path: articles/recent

Method: GET

Request Header:

{

“Authorization”: \_\_\_\_\_(String)

}

Response Body:

{

“status”: \_\_\_\_\_(Integer),

“msg”: \_\_\_\_\_(String),

“articles”: [

{

“pictUrl”:\_\_\_\_\_(String),

“path”: \_\_\_\_\_\_\_(String),

“title”: \_\_\_\_\_\_\_(String),

“fetchTime”:\_\_\_\_\_(Integer)

},

{

“pictUrl”:\_\_\_\_\_(String),

“path”: \_\_\_\_\_\_\_(String),

“title”: \_\_\_\_\_\_\_(String),

“fetchTime”: \_\_\_\_\_(Integer)

},

]

}

Status -> 0이면 성공

-> 1 이면 추천 실패

1. 관심 방문기: token -> [{제목, 주소, 태그, 좋아요를 눌렀는지, 방문기ID, 글쓴이 닉네임}]

쓰여진 날짜순으로 정렬해서 보낸다.

Content Type: application/json

Path: visits/heart

Method:GET

Request Header:

{

“Authorization”: \_\_\_\_\_(String)

}

Response Body:

{

“status”: \_\_\_\_\_(Integer),

“msg”: \_\_\_\_\_(String),

“visits”: [

{

“title”: \_\_\_\_\_\_\_ (String),

“writer”: \_\_\_\_\_\_(String),

“address”: \_\_\_\_\_(String),

“tags”: [“”, “”]

},

{

“title”: \_\_\_\_\_\_\_ (String),

“writer”: \_\_\_\_\_\_(String),

“address”: \_\_\_\_\_(String),

“tags”: [“”, “”]

}

]

}

Status -> 0이면 성공

-> 1 이면 조회 실패

쓰여진 날짜로 sort!

1. 자정마다 기사를 스크래이핑 -> 기사의 image, title, path, 기사ID 저장
2. 유저가 특정한 기사를 봤다는 것을 서버에게 알림: token, 기사 ID -> success or fail msg

Content Type: application/json

Path:articles/read/title/{title}/fetchTime/{fetchTime}

Method: POST

Request Header:

{

“Authorization”: \_\_\_\_\_(String)

}

Response Body:

{

“status”: \_\_\_\_\_(Integer),

“msg”: \_\_\_\_\_(String),

}

Status -> 0이면 성공

-> 1 이면 서버 처리 실패

1. 방문기 좋아요 -v

Content- type: application/json

Path: visits/heart/title/{title}/writer/{writer}

Method:POST

Request Header:

{

“Authorization”: \_\_\_\_\_(String)

}

Response Body:

{

“status”: \_\_\_\_\_,

“msg”: \_\_\_\_\_\_

}

1. 방문기 좋아요 취소 - v

Content- type: application/json

Path: visits/heart/title/{title}/writer/{writer}

Method:DELETE

Request Header:

{

“Authorization”: \_\_\_\_\_(String)

}

Response Body:

{

“status”: \_\_\_\_\_,

“msg”: \_\_\_\_\_\_

}

1. 유저 정보 가저오기:

Content- type: application/json

Path: user

Method:GET

Request Header:

{

“Authorization”: \_\_\_\_\_(String)

}

Response Body:

{

“status”: \_\_\_\_\_,

“msg”: \_\_\_\_\_\_,

“nickname”: \_\_\_\_\_,

“email”: \_\_\_\_\_,

}