□□□□ Pro Page□ □□□□ □□

> 1□1□□ (□□□ □□□)□ pro page□ □□□□

[□□□ □□□ - 1□1 □□ □□□□ □□]

1. Controller □□

`pro` □ `Controller` □ `dabory/themes/pro/app/Http/Controller/etc` □ `OneToOneController.php` □ □□□□ □□□□ □□□□□.

- □□ API □□ □□

□□□ API `App\Services\CallApiService` □ □□□□□.

```
class OneToOneController extends Controller
{
    private $callApiService;



    public function __construct(CallApiService $callApiService)
    {
        $this->callApiService = $callApiService;
    }



    public function list()
    {
        $limit = (int)request('limit', 12);
        $page = (int)request('page', 1);

        $oneToOnePage = $this->callApiService->callApi([
            'url' => 'list-type1-page',    // api□ □□□ url □□
            'data' => [                    // list-type1-page□ □□□ data
                'QueryVars' => [           // api □□□□ Query□ □□□ □ □□□ □□□ □□□
```

```
                'QueryName' => 'pro:my-page/post-list',   // pro/.../my-page/post-list/ □□□□ □□ □□ □□□ □□□.
                'SimpleFilter' => "post_code='1to1'",     // where □□ □□
                'SubSimpleFilter' => "image_type = 'thumb'",
                'IsntPagination' => false     // pagination □□
            ],
            'ListType1Vars' => [
                'OrderBy' => request('sort', 'mx.created_on desc')
            ],
            'PageVars' => [
                'Limit' => $limit,
                'Offset' => ($page - 1) * $limit
            ]
        ]
    ]);


//    dump($oneToOnePage);


    $oneToOnePage['Page'] = new LengthAwarePaginator($oneToOnePage['Page'],
$oneToOnePage['PageVars']['QueryCnt'],
        $limit, $page, ['path' => request()->url()]);
    // dump($oneToOnePage);
    return view('views.etc.1to1-list', compact('oneToOnePage'));
}


public function store()
{
    $validator = Validator::make(request()->all(), [
        'post_title' => 'required',
        'post_contents' => 'required',
        'pc1' => 'required',
        'pc2' => 'required|email'
    ]);


    if ($validator->fails()) {
        notify()->error(_e('Action failed'), 'Error', 'bottomRight');
        return redirect()->back()
```

```
            ->withErrors($validator)
            ->withInput();
        }


        $response =  $this->callApiService->callApi([
            'url' => 'post-act',
            'data' => [
                'Page' => [
                    [
                        'Id' => 0,
                        'PostTitle' => request('post_title'),
                        'PostContents' => request('post_contents'),
                        'PostTypeId' => 6,
                        'Pc1' => request('pc1'),
                        'Pc2' => request('pc2'),
                        'Status' => '2',
                        'MemberId' => session('member')['MemberId'],
                    ]
                ]
            ],
        ]);


        if ($this->callApiService->verifyApiError($response)) {
            notify()->error($response['body'], 'Error', 'bottomRight');
            return redirect()->back();
        }


//        notify()->success(_e('Action completed'), 'Success', 'bottomRight');
        return redirect()->route('1to1.list');
    }


    public function show($id)
    {
        $listType1Book = $this->callApiService->callApi([
            'url' => 'list-type1-book',
            'data' => [
```

```
        'Book' => [
            [
                'QueryVars' => [
                    'QueryName' => 'pro:my-page/post-details',
                    'SimpleFilter' => "post_code='1to1' and mx.id = $id",
                    'IsntPagination' => true,
                ],
                'PageVars' => [
                    'Limit' => 1
                ]
            ],
            [
                'QueryVars' => [
                    'QueryName' => 'pro:my-page/post-details-prenext',
                    'SimpleFilter' => "mx.id = (select max(id) from pro_post where id < ${id} and post_type_id =
6)",
                    'SubSimpleFilter' => "",
                    'IsntPagination' => true,
                ],
                'PageVars' => [
                    'Limit' => 1
                ]
            ],
            [
                'QueryVars' => [
                    'QueryName' => 'pro:my-page/post-details-prenext',
                    'SimpleFilter' => "mx.id = (select min(id) from pro_post where id > ${id} and post_type_id =
6)",
                    'SubSimpleFilter' => "",
                    'IsntPagination' => true,
                ],
                'PageVars' => [
                    'Limit' => 1
                ]
            ],
        ]
    ]
]);
```



```
    //    dd($listType1Book);
        if ($this->callApiService->verifyApiError($listType1Book)) {
            notify()->error($listType1Book['body'], 'Error', 'bottomRight');
            return redirect()->back();
        }



//        dump($listType1Book);
        return view('views.etc.1to1-details', compact('listType1Book'));
    }
}
```

1대1문의 구현을 위한 Controller 예시

이 `list-type1-page` URL로 data가 요청되`weberp-queries`에서 해당 URL의 QueryVars에 작성해준 조건에 해당하는 SQL 파일을 찾고, 데이터로 쿼리

응답 처리 및 **View** 반환

API로부터 응답을 받아 변수에 담고, View 페이지에서 필요한 데이터를 가공하거나 변환하여 반환합니다. View 페이지에서는 반환받은 배열 데이터를 통해 페이지 작업을 수행할 수 있습니다.

TEST

Controller와 view 페이지 작업이 완료되었다면 erp/pro 각각의 페이지에서 테스트를 진행합니다.

예제에서 진행했던 1대1문의 게시판을 확인해보겠습니다.

pro 페이지에 만들어준 문의하기 게시판에서 문의글을 작성한 뒤

게시판/블로그 관리
게시판 구분(공지/소개/문의)
공지형 게시판
소개형 게시판
문의형 게시판
상품문의
상담 신청
1대1 문의
공개 질문

erp 페이지로 이동하여 1대1문의 게시판을 확인합니다.



□□□□ □□□□□ □ □□□□□□.

□□□□

erp□ □□□ □□□□

erp □□□ □□□□□□

Revision #12
Created 17 June 2024 01:30:58 by Admin
Updated 10 July 2024 03:25:52 by □□□