Tutorial 4: Control Search and Filter Form

On the Post Home Screen, the Search and Filter forms are added by default. You can follow steps to control the Search form of each resource.

Step 1: Create PostSearchForm class to control search options

PostSearchForm.php

```
<?php
namespace Apps\Posts\Api\Form;
use Apps\Core_MobileApi\Api\Form\SearchForm;
class PostSearchForm extends SearchForm
{
        public function getSortOptions()
        {
                $sortOptions = parent::getSortOptions();
                $sortOptions[] = [
                'value' => 'is_featured',
                'label' => 'Is Featured'
                ];
                return $sortOptions;
        }
        public function getWhenOptions()
        {
                return parent::getWhenOptions();
        }
}
```

Step 2: Create Get Search From API

Now, let's improve **PostApi** class by adding the following code
PostApi.php

```
<?php
namespace Apps\Posts\Api\Service;
use Apps\Core_MobileApi\Adapter\MobileApp\MobileApp;
use Apps\Core_MobileApi\Adapter\MobileApp\MobileAppSettingInterface;
use Apps\Core_MobileApi\Adapter\MobileApp\Screen;
use Apps\Core_MobileApi\Adapter\Utility\UrlUtility;
use Apps\Core_MobileApi\Api\AbstractResourceApi;
use Apps\Posts\Api\Form\PostForm;
use Apps\Posts\Api\Form\PostSearchForm;
use Apps\Posts\Api\Resource\PostResource;


class PostApi extends AbstractResourceApi implements MobileAppSettingInterface
{
        public function __naming()
        {
                return [
                        'post/search-form' => [
                                'get' => 'getSearchForm'
                        ]
                ];
        }
        public function getSearchForm()
        {
                $form = $this->createForm(PostSearchForm::class, [
                        'title' => 'search',
                        'method' => 'GET',
                        'action' => UrlUtility::makeApiUrl('post')
                        ]);
                return $this->success($form->getFormStructure());
        }
        /* other code... */
}
```

Explain the code above:

- Method __**naming()** of **PostApi** class allows defining more API routes, Route to Search form MUST follow standard **"{resource_name}/search-form"**
- Method **getSearchForm()** handles API request and returns the response with the form's structure in JSON

Step 3: Configure Post Resource setting

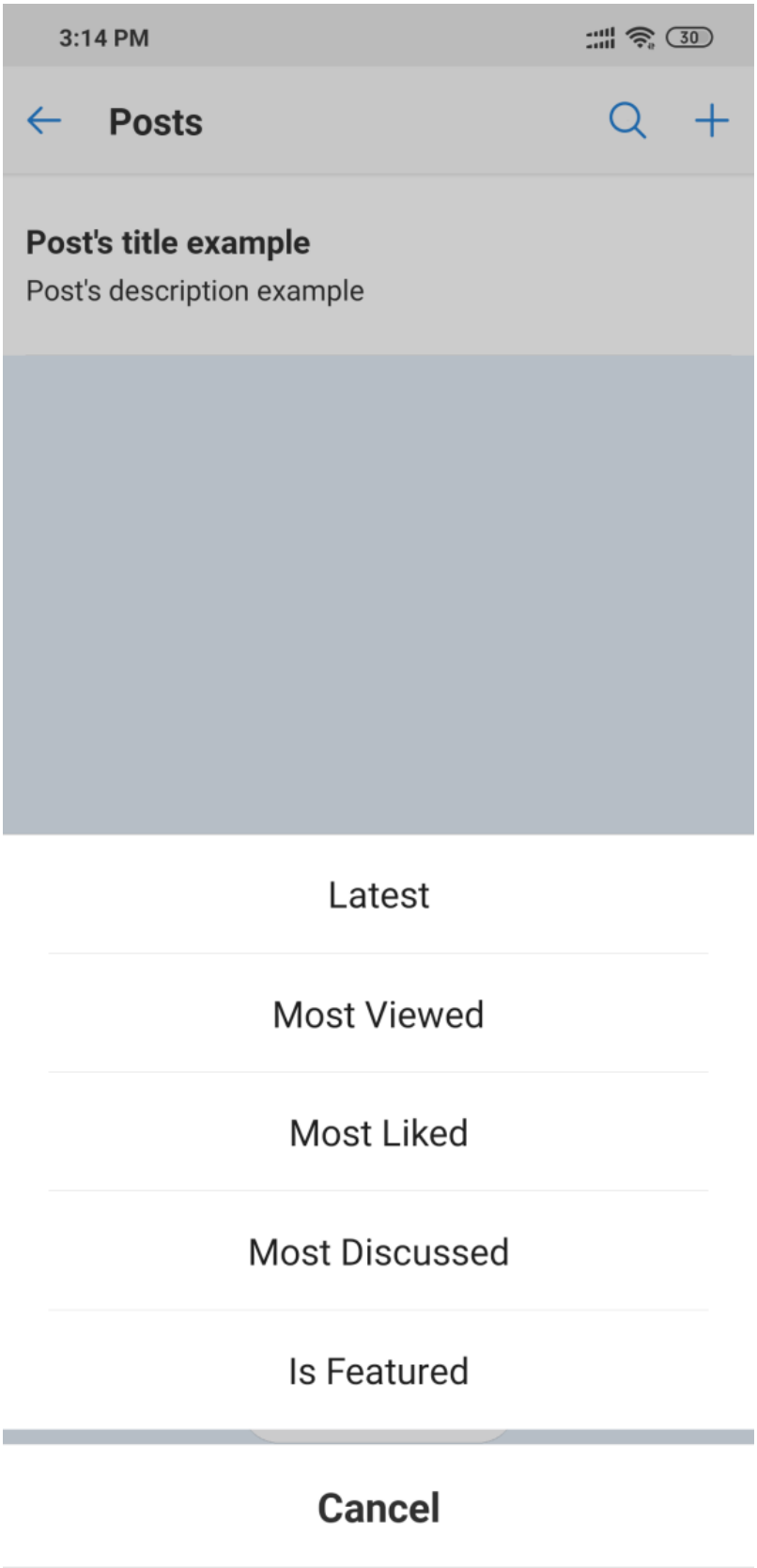
You can override method **getMobileSettings()** in **PostResource** class allows configuring each resource. The code below will change the placeholder string, modify sort and filter options
PostResource.php

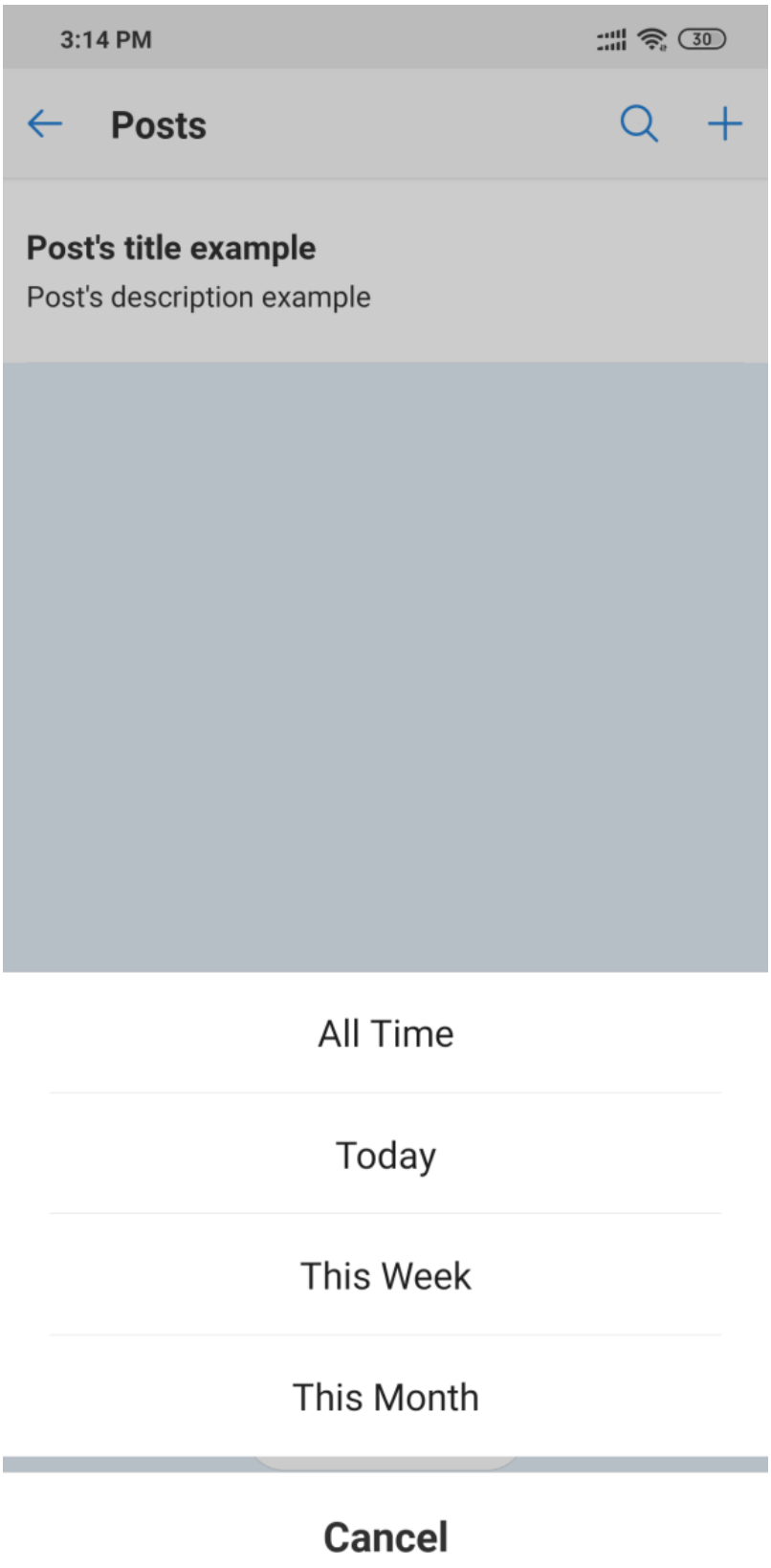
```
<?php
namespace Apps\Posts\Api\Resource;
use Apps\Core_MobileApi\Api\Resource\ResourceBase;
use Apps\Posts\Api\Form\PostSearchForm;


class PostResource extends ResourceBase
{
        public $resource_name = "post";
        public $module_name = "post";
        public $title;
        public $description;
        public $text;

        public function getMobileSettings($params = [])
        {
                $searchFilter = (new PostSearchForm());
                $searchFilter->setLocal($this->getLocalization());
                return self::createSettingForResource([
                        'resource_name' => $this->resource_name,
                        'search_input' => [
                                'placeholder' => 'Search Posts'
                        ],
                        'sort_menu' => $searchFilter->getSortOptions(),
                        'filter_menu' => $searchFilter->getWhenOptions()
                ]);
        }
}
```

Step 4: Test result on Mobile App


3:14 PM
30
Posts
Post's title example
Post's description example
Latest
Most Viewed
Most Liked
Most Discussed
Is Featured
Cancel



Step 5: Handle the search form submission

When a user uses the search or filter menu, the Mobile App will submit an API request to get All APIs with all form's parameters. All parameters are auto-resolved and passed to the **findAll()** method as an array.