ServiceArea ↔ UcmUser Link Model for Laravel + MongoDB

This document describes how to model a scalable many-to-many relationship between **ServiceArea** and **UcmUser** in Laravel using MongoDB, without storing large arrays on both sides of the relationship. Instead, a dedicated link (edge) collection is used.

1) Link Model (Pivot Equivalent in MongoDB)

```php
<?php
namespace App\Models;

use MongoDB\Laravel\Eloquent\Model;

class ServiceAreaUcmUser extends Model
{
    protected $collection = 'service_area_ucm_user_links';
    protected $fillable = ['service_area_id', 'ucm_user_id', 'meta', 'created_at', 'updated_at'];
    protected $casts = [
        'service_area_id' => 'objectid',
        'ucm_user_id'     => 'objectid',
        'meta'            => 'array',
    ];
}
```

2) ServiceArea Model

```php
<?php
namespace App\Models;

use MongoDB\Laravel\Eloquent\Model;

class ServiceArea extends Model
{
    protected $collection = 'service_areas';

    public function ucmUserLinks()
    {
        return $this->hasMany(ServiceAreaUcmUser::class, 'service_area_id', '_id');
    }

    public function ucmUsersQuery()
    {
        $userIds = ServiceAreaUcmUser::where('service_area_id', $this->_id)->pluck('ucm_user_id');
        return UcmUser::whereIn('_id', $userIds);
    }

    public function attachUsers(array $userIds, array $meta = [])
    {
        $userIds = array_values(array_unique($userIds));
        foreach (array_chunk($userIds, 1000) as $chunk) {
            foreach ($chunk as $uid) {
                ServiceAreaUcmUser::updateOrCreate(
                    ['service_area_id' => $this->_id, 'ucm_user_id' => $uid],
```

```
                    ['meta' => $meta, 'updated_at' => now()]
                );
            }
        }
    }

    public function detachUsers(array $userIds)
    {
        foreach (array_chunk($userIds, 5000) as $chunk) {
            ServiceAreaUcmUser::where('service_area_id', $this->_id)
                ->whereIn('ucm_user_id', $chunk)
                ->delete();
        }
    }

    public function syncUsers(array $newUserIds, array $meta = [])
    {
        $new     = collect($newUserIds)->unique()->values();
        $current = ServiceAreaUcmUser::where('service_area_id', $this->_id)->pluck('ucm_user_id');

        $toAdd    = $new->diff($current)->values()->all();
        $toRemove = $current->diff($new)->values()->all();

        $this->attachUsers($toAdd, $meta);
        $this->detachUsers($toRemove);
    }
}
```

3) UcmUser Model

```
<?php
namespace App\Models;

use MongoDB\Laravel\Eloquent\Model;

class UcmUser extends Model
{
    protected $collection = 'ucm_users';

    public function serviceAreaLinks()
    {
        return $this->hasMany(ServiceAreaUcmUser::class, 'ucm_user_id', '_id');
    }

    public function serviceAreasQuery()
    {
        $saIds = ServiceAreaUcmUser::where('ucm_user_id', $this->_id)->pluck('service_area_id');
        return ServiceArea::whereIn('_id', $saIds);
    }
}
```

4) MongoDB Indexes

```
$collection = (new ServiceAreaUcmUser())->getCollection();
$collection->createIndex(['service_area_id' => 1, 'ucm_user_id' => 1], ['unique' => true, 'name' => 'uni
$collection->createIndex(['ucm_user_id' => 1, 'service_area_id' => 1], ['name' => 'by_user_sa']);
```

5) Example Queries

```
// All ServiceAreas with at least one linked user
ServiceArea::whereHas('ucmUserLinks')->get();

// Areas that have a specific user
ServiceArea::whereHas('ucmUserLinks', fn($q) => $q->where('ucm_user_id', $userId))->get();

// Areas with counts
ServiceArea::withCount('ucmUserLinks')->orderByDesc('ucm_user_links_count')->paginate(50);

// Get users for one area
$serviceArea->ucmUsersQuery()->paginate(50);

// Get areas for one user
$ucmUser->serviceAreasQuery()->paginate(50);
```