Arsitektur Semart Skeleton

Muhamad Surya Iksanudin

Outline

- Instalasi dan Fitur Semart Skeleton
- Symfony dalam Kemudahan
- Annotation adalah Logic
- Kapan Menggunakan Annotation
- Kapan Membuat Custom Annotation
- Arsitektur Semart Skeleton
- Kenapa Harus Dipisahkan Per Use Case
- Service sebagai Business Logic
- Kapan Harus Membuat Service
- Repository sebagai Data Provider
- Hal-Hal yang Perlu Dihindari dalam Repository
- Memahami Konsep Entity Manager
- Controller sebagai Gateway
- Subscriber sebagai Hook

Instalasi Semart Skeleton

- Buka Terminal
- Jalankan git clone https://github.com/KejawenLab/SemartSkeleton.git Semart
- Pindah ke folder Semart
- Ubah file .env.dist menjadi .env dan sesuaikan konfigurasi databasenya
- Jalankan composer update --prefer-dist -vvv
- Jalan php bin/console semart:install
- Catat username dan password yang muncul pada layar terminal
- Jalankan php bin/console server:run
- Buka browser sesuai dengan alamat yang muncul pada layar terminal

Fitur-Fitur Utama Semart Skeleton

- Pengaturan User
- Manajemen Grup
- Pengaturan Menu
- Pemberian Hak Akses Sesuai Menu dan Grup
- Pengaturan Aplikasi
- CRUD Generator
- SQL Editor
- AdminLTE Template
- Pengurutan (Sorting)
- Pencarian (Searching)
- Manajemen Upload
- Ownership Filter

Belajar Symfony dari Semart Skeleton

- Belajar Lebih Fokus
- Langsung Praktek
- Use Case Berdasarkan Studi Kasus yang Real
- Tidak Perlu Belajar Semua Fitur Symfony
- Lebih Terstruktur
- Lebih Mudah Dipelajari

Fitur Utama Symfony pada Semart Skeleton

- Dependency Injection
- Event Dispatcher
- Security
- Annotation
- Validation
- Serializer
- Console/Command
- Translation
- Doctrine
- Twig

Bundle atau Library Lain yang Dipakai

- Knp Paginator Bundle (untuk Pagination)
- JS Routing Bundle (untuk Routing pada JS/Frontend)
- Doctrine Extension Bundle (untuk Timestampable dan Soft Deleteable)
- Ramsey UUID (untuk Primary Key)
- Snc Redis (untuk Session)
- Twine (untuk String Manipulation)
- Semart Collection (untuk Manipulasi Array)

Bundle dan Library Khusus pada Development

- Doctrine Fixtures
- PHP Unit
- Symfony Debug (Dump Variable maupun Object)
- Symfony Profiler (Debug Toolbar)
- Symfony Web Server

Dependency Injection

- Memasukkan Object ke Object Lain
- Biasanya Melalui Constructor
- Menyembunyikan Kompleksitas ketika Menginstansiasi Object
- Semakin Besar Aplikasi, Kompleksitas Object Semakin Meningkat

```php
<?php


$e = new E(new D(new C(new B(new A()))));
```

Event Dispatcher

- Merupakan Implementasi dari Observer Pattern
- Merubah Alur Program Menggunakan Source dari Luar
- Memudahkan Developer untuk Memanipulasi Program
- Menggunakan Event sebagai Cut Point

Event Dispatcher

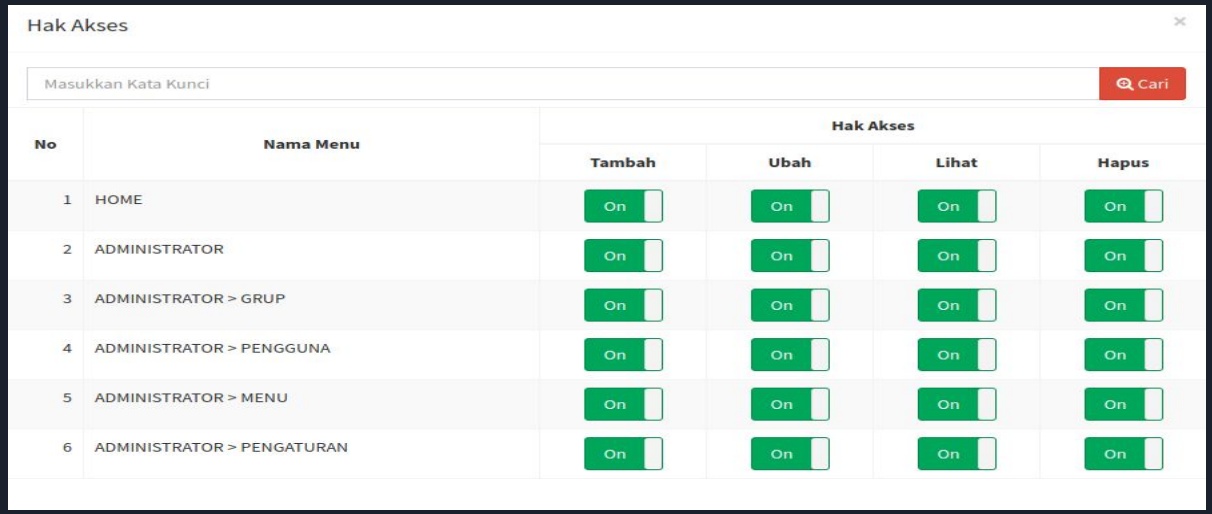
```php
$filterEvent = new RequestEvent($request, $object);
$this->eventDispatcher->dispatch( eventName: Application::REQUEST_EVENT, $filterEvent);
```

```php
class ExampleSubscriber implements EventSubscriberInterface
{
    public function filterRequest(RequestEvent $event)
    {
        // Kode Aplikasi yang La
    }

    public static function getSubscribedEvents()
    {
        return [
            Application::REQUEST_EVENT => [['filterRequest']],
        ];
    }
}
```

Security

- ➔ Menerapkan Sistem yang Lebih Mudah dan User Friendly
- ➔ Berbasis Grup dan Menu
- ➔ Menggunakan Annotation untuk Mempermudah
- ➔ Keamanan Sampai Level Per Action



Security

```
 * @Permission(menu="SETTING")
 *
 * @author Muhamad Surya Iksanudin <surya.iksanudin@gmail.com>
 */
class SettingController extends AdminController
{
    /**
     * @Route("/", methods={"GET"}, name="settings_index", options={"expose"=true})
     *
     * @Permission(actions=Permission::VIEW)
     */
    public function index(Request $request, Paginator $paginator)
    {
```

Annotation

- Metadata yang Terdapat dalam Source Code
- Di PHP Dibaca sebagai Comment oleh Interpreter (PHP Doc)
- Perlu Library untuk Membaca Annotation (Doctrine Annotation Reader)
- Annotation dapat Digunakan untuk Memanipulasi Alur secara Global
- Gunakan Annotation untuk Memanipulasi Logic yang Umum dan Terus-Menerus
- Gunakan Annotation untuk Sesuatu yang Jarang Diubah
- Digunakan pada Mapping Entity, Routing, Validation, Serializer, Security (Permission), Sorting dan Searching (Mapping)

Translation

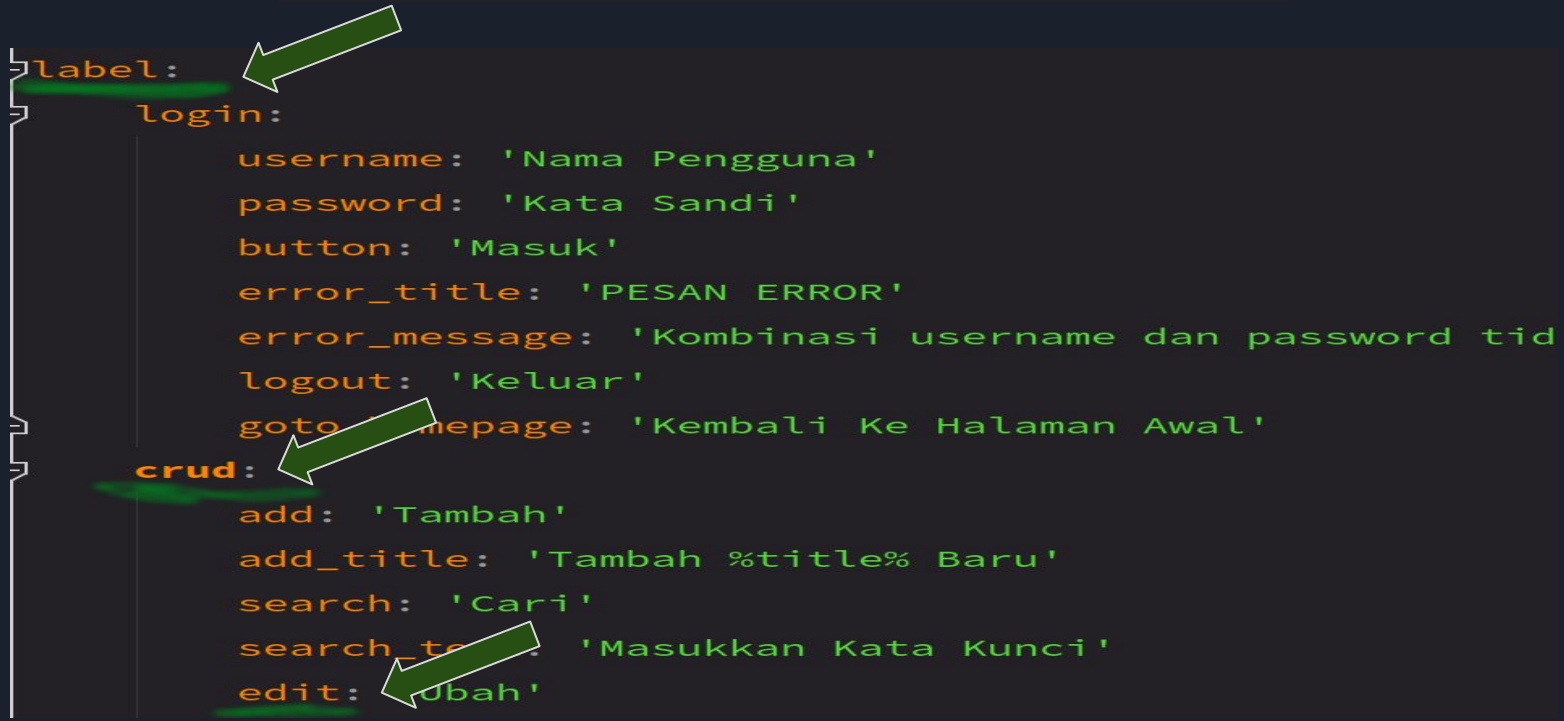
- ➔ Berada di Folder translations/messages.<lang>.yaml
- ➔ Digunakan untuk Menerjemahkan Label ke Bahasa Tertentu
- ➔ Gunakan File Yaml untuk Memudahkan

```
$this->translator->trans( id: 'label.yang_ditranslate');
```

```
{{ 'label.crud.edit' | trans }}
```

Translation

```
{{ 'label.crud.edit' | trans }}
```



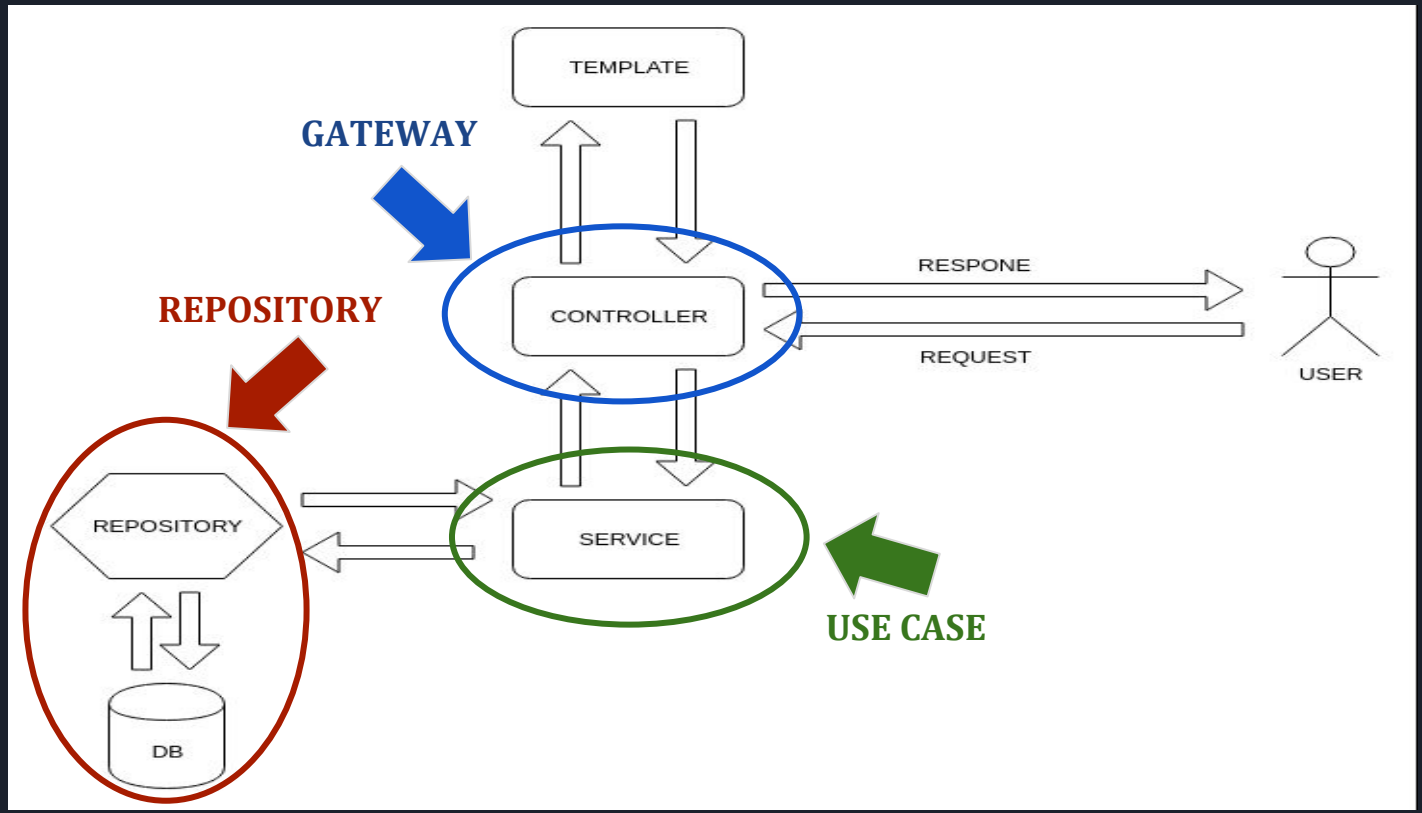
Twig

- Template Engine untuk Symfony
- Terinspirasi dari Jinja pada Python
- Aman dan Cepat
- Mudah Dipelajari
- Bersih

Twig

```
{% for key, data in users %}
    <tr>
        <td class="text-right">{{ (key + startNumber)  }}</td>
        <td>{{ data.fullName }}</td>
        <td>{{ data.username }}</td>
        <td>{{ data.group.name }}</td>
        <td class="text-center">
            {% if is_granted('edit', menu) and false == data.del
                <button data-primary="{{ data.id }}" class="btn
            {% endif %}
            {% if is_granted('delete', menu) and 'admin' != data
                <button data-primary="{{ data.id }}" class="btn
            {% endif %}
        </td>
    </tr>
{% endfor %}
```

Arsitektur Semart Skeleton



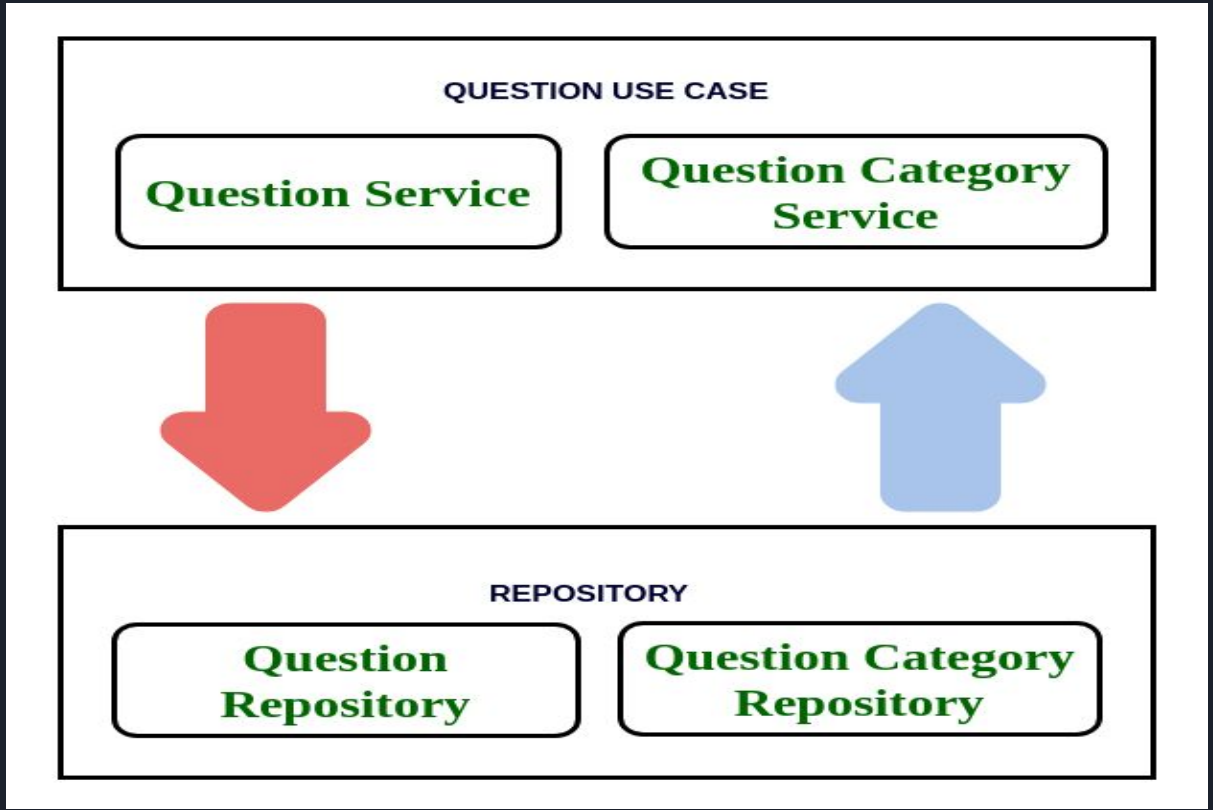
Apa itu Use Case

- ➔ Permasalahan yang Telah Didefinisikan
- ➔ Hanya Fokus pada Satu Lingkup Permasalahan
- ➔ Tidak Memperdulikan Interaksi dengan Database
- ➔ Fokus pada Solving
- ➔ Berinteraksi dengan Data Melalui Layer Repository
- ➔ Berinteraksi dengan Interface

Contoh Use Case (Soal Ujian)

- ➔ Kategori Soal
- ➔ Soal
- ~~➔ Jawaban~~
- ~~➔ Ujian~~
- ~~➔ Peserta~~
- ~~➔ Jawaban Ujian~~

Use Case


QUESTION USE CASE
Question Service
Question Category Service
REPOSITORY
Question Repository
Question Category Repository

Use Case

- Service sebagai Business Logic per Masalah
- Repository sebagai Data Provider
- Satu Service dapat Berinteraksi dengan Satu atau Lebih Repository
- Satu Service Bisa Berinteraksi dengan Service Lain di Use Case Lainnya
- Penamaan Service Harus Spesifik dan Deskriptif

Kenapa Harus Per Use Case

- ➔ Lebih Fokus
- ➔ Permasalahan Lebih Sederhana
- ➔ Mudah dalam Mendesign Solusi
- ➔ Lebih Memudahkan dalam Berfikir

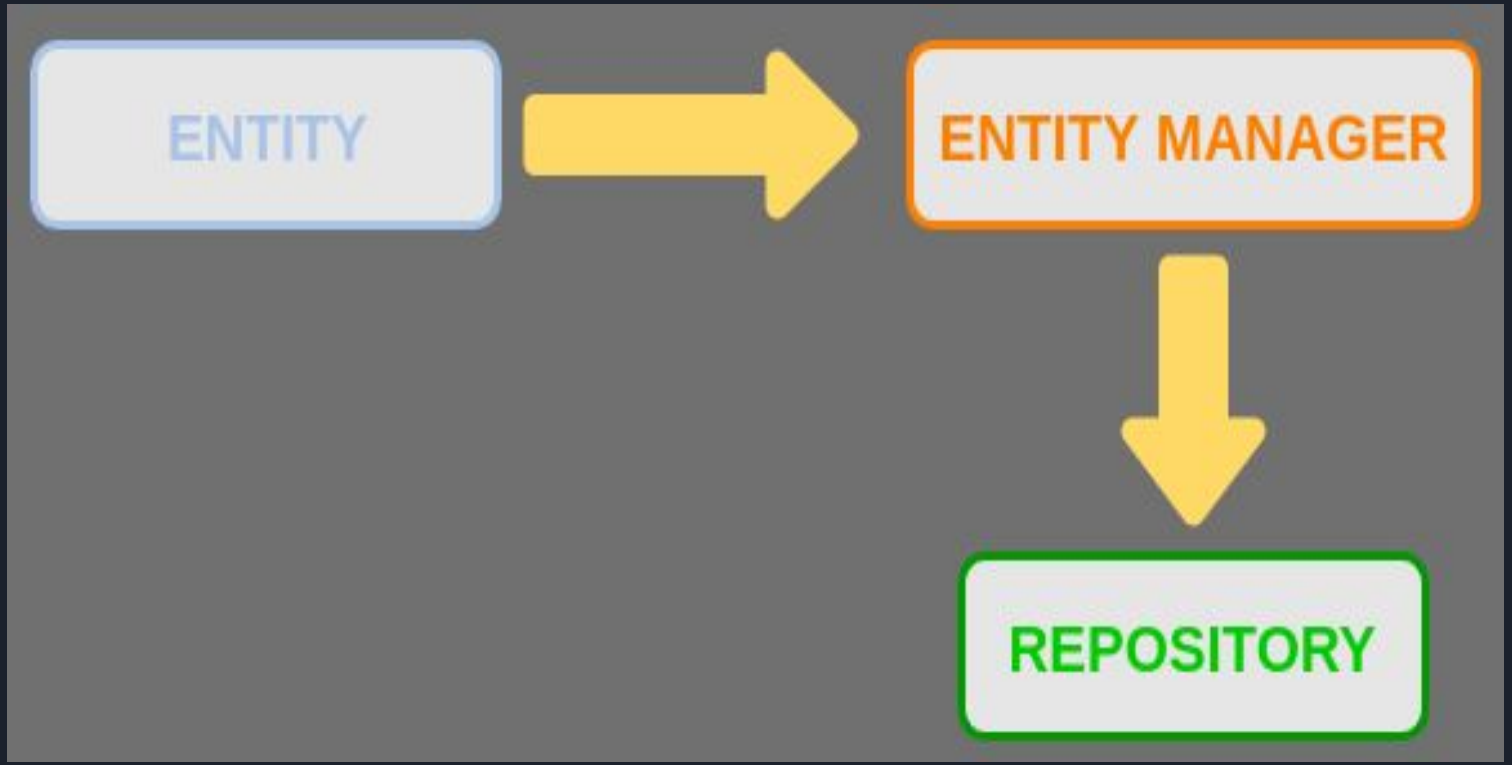
Repository

- Data Provider yang Berhubungan dengan Database
- Repository HANYA Berhubungan dengan Service dan Database
- Query Database Dibuat Sesimpel Mungkin
- Gunakan Cache untuk Interaksi Antar Repository dalam Service
- Hindari Query yang Kompleks
- Usahakan Format Data Keluaran Sebagai Object atau Array Object

Doctrine

- Object Relational Model (ORM) di PHP
- Enterprise Level Library
- Support Mayor Database
- Support No SQL Melalui ODM
- Memisahkan Antara Entity (Model), Entity Manager (Provider) dan Repository
- Config Menggunakan Annotation
- Disupport oleh Symfony

Arsitektur Doctrine



Permasalah yang Mungkin Muncul pada Doctrine

Exceptions

A new entity was found through the relationship 'KejawenLab\Semart\Skeleton\Entity\Answer#question' that was not configured to cascade persist operations for entity: KejawenLab\Semart\Skeleton\Entity\Question@0000000049b2555400000000013354056. To solve this issue: Either explicitly call EntityManager#persist() on this unknown entity or configure cascade persist this association in the mapping for example @ManyToOne(..,cascade={"persist"}). If you cannot find out which entity causes the problem implement 'KejawenLab\Semart\Skeleton\Entity\Question#__toString()' to get a clue.

- ➔ Terjadi karena property $question berisi object yang belum di-manage oleh Doctrine
- ➔ Terjadi karena object diambil dari cache sehingga tidak melalui mekanisme Doctrine
- ➔ Bisa juga karena object $question memang benar-benar baru

Solusi

Exceptions

A new entity was found through the relationship 'KejawenLab\Semart\Skeleton\Entity\Answer#question' that was not configured to cascade persist operations for entity: KejawenLab\Semart\Skeleton\Entity\Question@0000000049b2555400000000013354056. To solve this issue: Either explicitly call EntityManager#persist() on this unknown entity or configure cascade persist this association in the mapping for example @ManyToOne(..,cascade={"persist"}). If you cannot find out which entity causes the problem implement 'KejawenLab\Semart\Skeleton\Entity\Question#__toString()' to get a clue.

- ➔ Persist object $question ($em->persist($question);)
- ➔ Bila yakin Object Baru bisa menambahkan config cascade pada entity
- ➔ Merge object $question ($em->merge($question);)

TERIMA KASIH