Creating model behaviors

There are many similar solutions in today’s web applications. Leading products such as Google’s Gmail are defining nice UI patterns. One of these is soft delete. Instead of a permanent deletion with tons of confirmations, Gmail allows us to immediately mark messages as deleted and then easily undo it. The same behavior can be applied to any object such as blog posts, comments, and so on.

Let’s create a behavior that will allow marking models as deleted, restoring models, selecting not yet deleted models, deleted models, and all models. In this recipe, we’ll follow a test-driven development approach to plan the behavior and test if the implementation is correct.

Getting ready

1. Create a new yii2-app-basic application using the composer as described in the official guide at <http://www.yiiframework.com/doc-2.0/guide-start-installation.html>.

2. Create two databases for working and for tests.

3. Configure Yii to use the first database in your primary application in config/db. php. Make sure the test application uses the second database in tests/codeception/config/config. php.

4. Create a new migration:

<?php

use yii\db\Migration;

class m160427\_103115\_create\_post\_table extends Migration {

public function up()

{

$this->createTable('{{%post}}', [

'id' => $this->primaryKey(),

'title' => $this->string()->notNull(),

'content\_markdown' => $this->text(),

'content\_html' => $this->text(),

]);

}

public function down()

{

$this->dropTable('{{%post}}');

}

}

5. Apply the migration to both the working and test databases:

./yii migrate

tests/codeception/bin/yii migrate

6. Create the Post model:

<?php

namespace app\models;

use app\behaviors\MarkdownBehavior; use yii\db\ActiveRecord;

/\*\*