Лабораторная работа №2

Работа с базой данных

Для работы с базой данных в Yii необходимо настроить подключение к базе данных, создать класс Active Record, определить action и создать view.

Для начала работы необходимо наличие базы данных, например, с именем yii2basic. Представим, что в этой базе есть таблица country, содержащая демонстрационные данные.

```
CREATE TABLE `country` (
  `code` CHAR(2) NOT NULL PRIMARY KEY,
  `name` CHAR(52) NOT NULL,
  `population` INT(11) NOT NULL DEFAULT '0'
) ENGINE=InnoDB DEFAULT CHARSET=utf8;

INSERT INTO `country` VALUES ('AU','Australia',24016400);
INSERT INTO `country` VALUES ('BR','Brazil',205722000);
INSERT INTO `country` VALUES ('CA','Canada',35985751);
INSERT INTO `country` VALUES ('CN','China',1375210000);
INSERT INTO `country` VALUES ('DE','Germany',81459000);
INSERT INTO `country` VALUES ('FR','France',64513242);
INSERT INTO `country` VALUES ('GB','United Kingdom',65097000);
INSERT INTO `country` VALUES ('IN','India',1285400000);
INSERT INTO `country` VALUES ('RU','Russia',146519759);
INSERT INTO `country` VALUES ('US','United States',322976000);
```

Настройка подключение к БД

Для начала необходимо убедиться, что установлены PHP-расширение PDO и драйвер PDO для используемой базы данных (например, pdo_mysql для MySQL). Это базовое требование в случае использования вашим приложением реляционной базы данных. После того, как они установлены, откройте файл config/db.php и измените параметры на верные для вашей базы данных. По умолчанию этот файл содержит следующее:

```
<?php
return [
    'class' => 'yii\db\Connection',
```

```
    'dsn' => 'mysql:host=localhost;dbname=yii2basic',
    'username' => 'root',
    'password' => '',
    'charset' => 'utf8',
];
```

Файл config/db.php — типичный конфигурационный инструмент, базирующийся на файлах. Данный конфигурационный файл определяет параметры, необходимые для создания и инициализации экземпляра yii\db\Connection, через который вы можете делать SQL запросы к подразумеваемой базе данных.

Подключение к БД, настроенное выше, доступно в коде приложения через выражение Yii::$app->db.

Информация: файл config/db.php будет подключен главной конфигурацией приложения config/web.php, описывающей то, как экземпляр приложения должен быть инициализирован.

Создание модели потомка Active Record

Чтобы представлять и получать данные из таблицы country, создайте класс — потомок Active Record, под названием Country и сохраните его в файле models/Country.php.

```
<?php
namespace app\models;
use yii\db\ActiveRecord;
class Country extends ActiveRecord
{
//код класса
}
```

Класс Country наследуется от yii\db\ActiveRecord. Внутри него не нужно писать ни строчки кода. С кодом, приведённым выше, Yii свяжет имя таблицы с именем класса.

Информация: Если нет возможности задать прямую зависимость между именем таблицы и именем класса, вы можете переопределить метод yii\db\ActiveRecord::tableName(), чтобы явно задать имя связанной таблицы.

Используя класс Country, вы можете легко манипулировать данными в таблице country, как показано в этих фрагментах:

```php
use app\models\Country;
// получаем все строки из таблицы "country" и сортируем их по "name"
$countries = Country::find()->orderBy('name')->all();
// получаем строку с первичным ключом "US"
$country = Country::findOne('US');
// отобразит "United States"
echo $country->name;
// меняем имя страны на "U.S.A." и сохраняем в базу данных
$country->name = 'U.S.A.';
$country->save();
```

Информация: Active Record — мощный способ доступа и манипулирования данными БД в объектно-ориентированном стиле. Вы можете найти подробную информацию в разделе Active Record. В качестве альтернативы, вы также можете взаимодействовать с базой данных, используя более низкоуровневый способ доступа, называемый Data Access Objects.

Создание Action

Для того, чтобы показать данные по странам конечным пользователям необходимо создать контроллер, например CountryController, а в нем новый action index.

```php
<?php
namespace app\controllers;
use yii\web\Controller;
use yii\data\Pagination;
use app\models\Country;
class CountryController extends Controller
{
    public function actionIndex()
    {
        $query = Country::find();
        $pagination = new Pagination([
            'defaultPageSize' => 5,
            'totalCount' => $query->count(),
        ]);
        $countries = $query->orderBy('name')
            ->offset($pagination->offset)
            ->limit($pagination->limit)
            ->all();
```

```
        return $this->render('index', [
            'countries' => $countries,
            'pagination' => $pagination,
        ]);
    }
}
```

Action index вызывает Country::find(). Данный метод Active Record строит запрос к БД и извлекает все данные из таблицы country. Чтобы ограничить количество стран, возвращаемых каждым запросом, запрос разбивается на страницы с помощью объекта yii\data\Pagination. Объект Pagination служит двум целям:

- Устанавливает пункты offset и limit для SQL инструкции, представленной запросом, чтобы она возвращала только одну страницу данных за раз (в примере максимум 5 строк на страницу).
- Он используется во view для отображения пагинатора, состоящего из набора кнопок с номерами страниц, это будет разъяснено в следующем подразделе.

В конце кода action index выводит view с именем index, и передаёт в него данные по странам вместе с информацией о пагинации.

Создание View

Первым делом необходимо создать поддиректорию с именем country внутри директории views. Эта папка будет использоваться для хранения всех view, выводимых контроллером country. Внутри директории views/country создайте файл с именем index.php, содержащий следующий код:

```
<?php
use yii\helpers\Html;
use yii\widgets\LinkPager;
?>
<h1>Countries</h1>
<ul>
<?php foreach ($countries as $country): ?>
    <li>
        <?= Html::encode("{$country->code} ({$country->name})") ?>:
        <?= $country->population ?>
    </li>
<?php endforeach; ?>
```

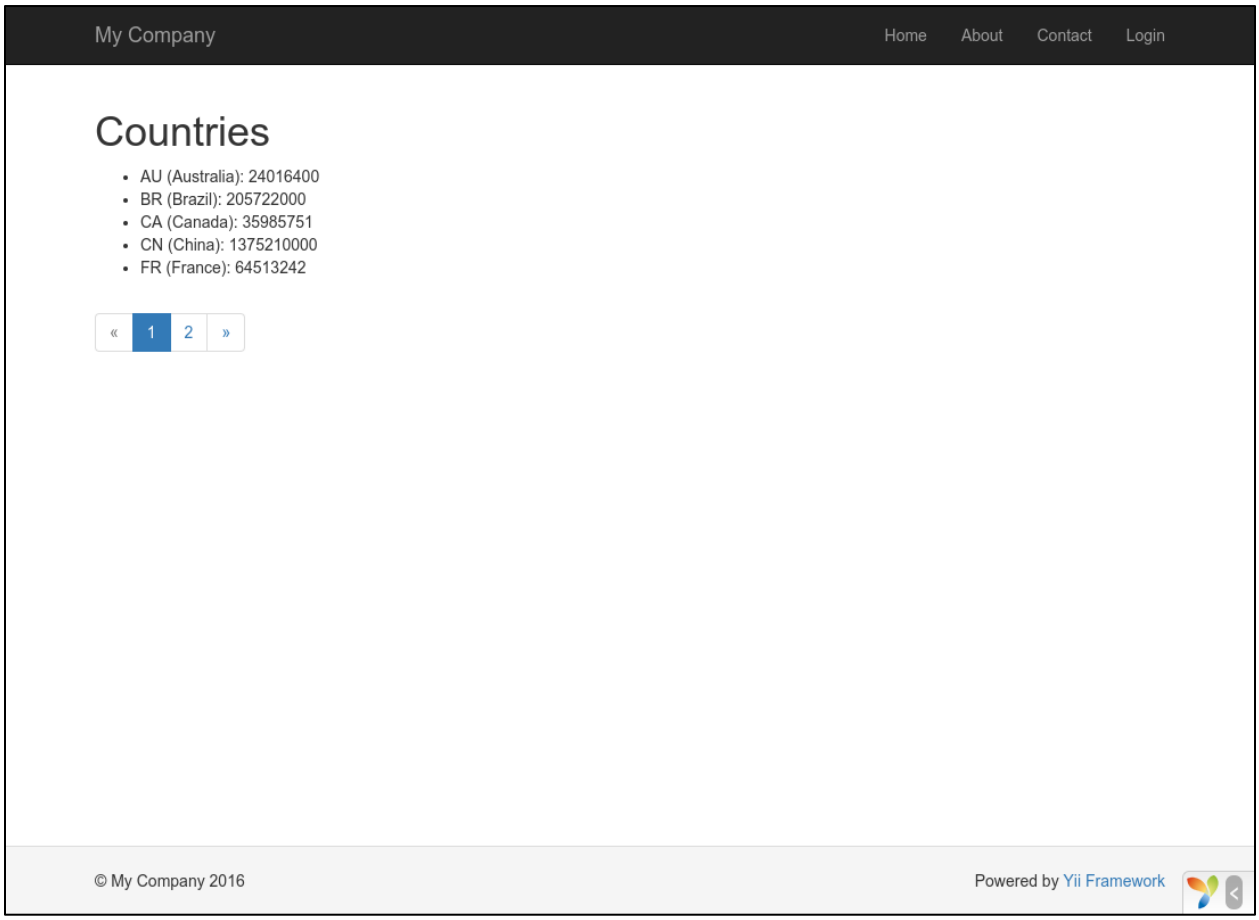
```
</ul>
<?= LinkPager::widget(['pagination' => $pagination]) ?>
```

View имеет 2 части относительно отображения данных по странам. В первой части предоставленные данные по странам выводятся как неупорядоченный HTML-список. Во второй части выводится виджет yii\widgets\LinkPager, используя информацию о пагинации, переданную из action во view. Виджет LinkPager отображает набор постраничных кнопок. Клик по любой из них обновит данные по странам в соответствующей странице.

Тестирование примера

Чтобы увидеть, как работает весь вышеприведённый код, перейдите по следующей ссылке в своём браузере:

http://hostname/index.php?r=country%2Findex



В начале вы увидите страницу, показывающую пять стран. Под странами вы увидите пагинатор с четырьмя кнопками. Если вы кликните по кнопке "2", то увидите

страницу, отображающую другие пять стран из базы данных: вторая страница записей. Посмотрев внимательней, вы увидите, что URL в браузере тоже сменилось на http://hostname/index.php?r=country%2Findex&page=2

За кадром Pagination предоставляет всю необходимую функциональность для постраничной разбивки набора данных:

- В начале Pagination показывает первую страницу, которая отражает SELECT запрос стран с параметрами LIMIT 5 OFFSET 0. Как результат, первые пять стран будут получены и отображены.
- Виджет LinkPager выводит кнопки страниц используя URL'ы, созданные Pagination. Эти URL'ы будут содержать параметр запроса page, который представляет различные номера страниц.
- Если вы кликните по кнопке "2", сработает и обработается новый запрос для маршрута country/index. Таким образом новый запрос стран будет иметь параметры LIMIT 5 OFFSET 5 и вернет следующие пять стран для отображения.

Задания к лабораторной работе

Максимальный балл за лабораторную работу – 6 баллов
Итоговый балл может быть понижен в следующих случаях: 1) задания выполнены некорректно, не в полном объеме, содержат ошибки, реализованы недостаточно оптимально и т.п.; 2) студент не может обосновать и аргументированно ответить на вопросы по заданиям, не владеет теоретическим материалом и т.п.

Для выполнения лабораторной работы вам необходима база данных библиотеки.

Задание 1

- Настройте подключение к базе данных.
- Создайте модель- потомок Active Record для работы с таблицей авторы.
- Измените код контроллера для взаимодействия с моделью, формирования запросов к базе данных и работы с нужным представлением.
- Создайте представление для работы с авторами, которое позволяет отобразить в табличном виде данные таблицы автор.
- Протестируйте работоспособность примера.

Задание 2

- Создайте модель и представление для работы с таблицей жанры. Выведите содержимое таблицы.
- Создайте модель и представление для работы с таблицей книги. Выполните соединение таблиц для отображения связанных данных из таблиц жанр и автор. Выведите содержимое таблицы (выводится не id жанра и id автора, а их названия/имена)

  Для связывания данных в модели вам могут понадобиться методы hasOne, hasMany.

Задание 3

Выведите результат выполнения следующих запросов:

- Найти книги, написанные в 20 веке. Отсортировать по году написания.
- Вывести авторов и количество написанных ими книг.
- Найти книги, в названии которых содержится слово. Слово вводить в форму.

  Для выполнения запросов вам могут понадобиться методы построителя запросов: where, orderby, groupby, select, with, find, findone.

Задание 4

Создайте форму для добавления нового автора. Выполните добавление введенных данных в таблицу.

Используйте метод save.

Задание 5

Создайте форму для удаления автора. Введите в форму id автора и удалите его из таблицы.

Используйте метод delete.

Контрольные вопросы

1. В каком файле выполняется подключение к базе данных?
2. Какие методы отвечают за добавление данных в базу данных?
3. Какой метод позволяет ограничить количество возвращаемых строк данных?
4. В чем особенности жадной загрузки данных?