МИНИСТЕРСТВО ОБРАЗОВАНИЯ И НАУКИ РФ

Федеральное государственное бюджетное образовательное учреждение

высшего образования

«Белгородский государственный технологический университет

им. В.Г. Шухова»

**ИНСТИТУТ ЭНЕРГЕТИКИ, ИНФОРМАЦИОННЫХ ТЕХНОЛОГИЙ**

**И УПРАВЛЯЮЩИХ СИСТЕМ**

**Кафедра программного обеспечения вычислительной**

**техники и автоматизированных систем**

Лабораторная работа №3

по дисциплине: Технологии web-программирования

на тему:

**Серверное программирование. Yii2**

Выполнил:

ст. гр. ПВ-42

Шошин О.В.

Проверил:

Картамышев С.В.

Белгород 2020

**Задание к лабораторной работе:**

1. Развернуть базовое приложение Yii2 App Basic.
2. Настроить конфигурацию работы приложения с docker.
3. Добавить модуль для работы с API.
4. Добавить несколько контроллеров со статическими данными.
5. Продемонстрировать работу API в Postman.

**Выполнение**

* + - 1. Развернул базовое приложение с Yii2 App Basic с помощью команды:

docker-compose run php composer create-project --prefer-dist yiisoft/yii2-app-basic basic

Переместил проект в рабочую папку.

Сконфигурировал приложение:

В файл app\config\db.php внес информацию о БД:

<?php

return [

    'class' => 'yii\db\Connection',

    'dsn' => 'mysql:host=mysql;dbname=bstu',

    'username' => 'bstu',

    'password' => 'bstu321!',

    'charset' => 'utf8',

];

В файле app\config\web.php добавил секцию urlManager:

   'urlManager' => [

            'enablePrettyUrl' => true,

            'showScriptName' => false,

            'rules' => [

            ],

        ],

]

* + - 1. Настроил конфигурацию работы приложения с docker.

Заполнил файл docker-compose.yml для конфигурации контейнеров docker:

version: '3.1'

services:

  php:

    container\_name: car\_trade\_php

    image: yiisoftware/yii2-php:7.3-apache

    volumes:

      - ~/.composer-docker/cache:/root/.composer/cache:delegated

      - ./:/app:delegated

    ports:

      - '1199:80'

  mysql:

    container\_name: car\_trade\_mysql

    image: mysql

    command: --default-authentication-plugin=mysql\_native\_password --character-set-server=utf8 --collation-server=utf8\_general\_ci

    environment:

      - MYSQL\_ROOT\_PASSWORD=verysecret

      - MYSQL\_DATABASE=bstu

      - MYSQL\_USER=bstu

      - MYSQL\_PASSWORD=bstu321!

Запустил контейнеры с помощью команды:

docker-compose up -d

* + - 1. Добавил модуль работы с API.

Модуль v1 находится в папке app\modules\v1.

Модуль описывается в файле app\module\v1\v1.php в классе v1. Класс v1 наследуется от класса Module(yii\base\Module):

<?php

namespace app\modules\v1;

use yii\base\Module;

class v1 extends Module

{

}

Чтобы сообщить приложению о модуле, необходимо добавить информацию о модуле в файл web.php:

    'modules' => [

        'v1' => [

            'class' => 'app\modules\v1\v1'

        ]

    ]

* + - 1. Создал базовый класс контроллера ApiController.

Добавил функциональную возможность выполнения кросс-доменных запросов для дочерних запросов и удалил отправку ответа запроса в форме XML.

class ApiController extends Controller

{

    public function behaviors()

    {

        return [

            'contentNegotiator' => [

                'class' => ContentNegotiator::className(),

                'formats' => [

                    'application/json' => Response::FORMAT\_JSON

                ],

            ],

            'verbFilter' => [

                'class' => VerbFilter::className(),

                'actions' => $this->verbs(),

            ],

            'authenticator' => [

                'class' => CompositeAuth::className(),

            ],

            'rateLimiter' => [

                'class' => RateLimiter::className(),

            ],

            'corsFilter' => [

                'class' => Cors::class

            ]

        ];

    }

}

Создал класс CarController, который наследуется от ApiController.

Добавил один action – actionIndex, который возвращает массив с данными об автомобилях.

<?php

namespace app\modules\v1\controllers;

class CarController extends ApiController

{

    public function actionIndex()

    {

        $cars = [

            [

                'id' => 1,

                'name' => 'Skoda Rapid',

                'year' => 2012,

                'price' => 449000,

                'odometer' => 74000,

                'volume' => 1.6,

                'power' => 102,

                'fuel' => 'Бензин',

                'drivetrain' => 'Передний',

                'color' => 'Белый',

                'transmission' => 'Механика',

                'bodyStyle' => 'Седан'

            ],

            [

                'id' => 2,

                'name' => 'Skoda Rapid',

                'year' => 2019,

                'price' => 888000,

                'odometer' => 200000,

                'volume' => 1.6,

                'power' => 102,

                'fuel' => 'Бензин',

                'drivetrain' => 'Передний',

                'color' => 'Белый',

                'transmission' => 'Автомат',

                'bodyStyle' => 'Седан'

            ],

            [

                'id' => 3,

                'name' => 'Skoda Rapid',

                'year' => 2015,

                'price' => 889000,

                'odometer' => 600000,

                'volume' => 1.6,

                'power' => 102,

                'fuel' => 'Газ',

                'drivetrain' => 'Передний',

                'color' => 'Белый',

                'transmission' => 'Автомат',

                'bodyStyle' => 'Седан'

            ]

        ];

        return $cars;

    }

}

* + - 1. Демонстрация работы в Postman.

