**Министерство науки и высшего образования РФ**

**Государственное образовательное учреждение высшего профессионального образования**

**Белгородский Государственный Технологический Университет им В. Г. Шухова**

Кафедра программного обеспечения вычислительной техники и

автоматизированных систем.

Лабораторная работа №3 по дисциплине

«Технологии Web-программирования»

Тема: Серверное программирование. Yii2

Выполнил:

студент группы ПВ-41

Колесников В.Д.

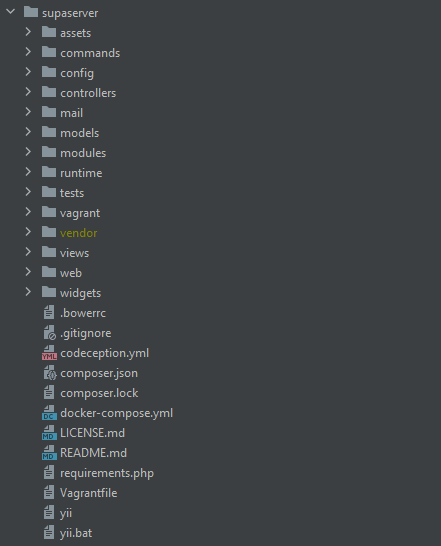
Проверил: Поляков В.К.

Белгород 2019 г.

**Цель работы:** познакомиться с основами backend разработки web-приложений. Научится писать скрипты на языке PHP. Познакомиться с основами работы docker. Познакомиться с фреймворком Yii2 и научиться разворачивать проект, производить его настройку. Научится работать с API в приложении Postman.

**Выполение**

Было развёрнуто базовое приложение Yii2 с помощью php-композера командой composer create-project --prefer-dist yiisoft/yii2-app-basic basic.



Далее был настроен файл конфигурации docker-compose.yml

version: '3.1'

services:

php:

image: yiisoftware/yii2-php:7.3-apache

volumes:

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

- ./:/app:delegated

ports:

- '1199:80'

extra\_hosts:

- "visiticelandbeauty.com:127.0.0.1"

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=db

- MYSQL\_USER=user

- MYSQL\_PASSWORD=password

Был добавлен API модуль версии 1 со следующими контроллерами и моделями:

**User.php**

<?php

namespace app\models;

class User extends \yii\base\BaseObject implements \yii\web\IdentityInterface

{

public $id;

public $username;

public $password;

public $authKey;

public $accessToken;

private static $users = [

'100' => [

'id' => '100',

'username' => 'admin',

'password' => 'admin',

'authKey' => 'test100key',

'accessToken' => '100-token',

],

'101' => [

'id' => '101',

'username' => 'demo',

'password' => 'demo',

'authKey' => 'test101key',

'accessToken' => '101-token',

],

];

/\*\*

\* {@inheritdoc}

\*/

public static function findIdentity($id)

{

return isset(self::$users[$id]) ? new static(self::$users[$id]) : null;

}

/\*\*

\* {@inheritdoc}

\*/

public static function findIdentityByAccessToken($token, $type = null)

{

foreach (self::$users as $user) {

if ($user['accessToken'] === $token) {

return new static($user);

}

}

return null;

}

/\*\*

\* Finds user by username

\*

\* @param string $username

\* @return static|null

\*/

public static function findByUsername($username)

{

foreach (self::$users as $user) {

if (strcasecmp($user['username'], $username) === 0) {

return new static($user);

}

}

return null;

}

/\*\*

\* {@inheritdoc}

\*/

public function getId()

{

return $this->id;

}

/\*\*

\* {@inheritdoc}

\*/

public function getAuthKey()

{

return $this->authKey;

}

/\*\*

\* {@inheritdoc}

\*/

public function validateAuthKey($authKey)

{

return $this->authKey === $authKey;

}

/\*\*

\* Validates password

\*

\* @param string $password password to validate

\* @return bool if password provided is valid for current user

\*/

public function validatePassword($password)

{

return $this->password === $password;

}

}

**ApiController.php**

<?php

namespace app\modules\v1\controllers;

use yii\filters\auth\CompositeAuth;

use yii\filters\ContentNegotiator;

use yii\filters\Cors;

use yii\filters\RateLimiter;

use yii\filters\VerbFilter;

use yii\rest\Controller;

use yii\web\Response;

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

]

];

}

}

**EmotesController.php**

<?php

namespace app\modules\v1\controllers;

class EmotesController extends ApiController {

public function actionHome() {

$emotes = [

[

'id'=> 1,

'name' => 'Okayga',

'description' => 'FeelsOkayMan + Pepega.',

'image' => '@/assets/images/Okayga.png',

],

[

'id' => 2,

'name' => 'PeepoClown',

'description' => 'Because twitch viewers are often clowns.',

'image' => '@/assets/images/PeepoClown.png',

],

[

'id' => 3,

'name' => 'forsenCD',

'description' => 'D OMEGALUL C emote.',

'image' => '@/assets/images/forsenCD.jpg',

],

[

'id' => 4,

'name' => 'FeelsAmazingMan',

'description' => 'When you are totally happy.',

'image'=> '@/assets/images/FeelsAmazingMan.png',

],

[

'id' => 5,

'name' => 'Pepega',

'description' => 'Absolute retardness.',

'image' => '@/assets/images/Pepega.png',

],

[

'id' => 6,

'name' => 'forsenE',

'description' => 'The face of Twitch.',

'image' => '@/assets/images/forsenE.jpg',

],

[

'id' => 7,

'name' => 'KKomrade',

'description' => 'Fellow russians emote',

'image' => '@/assets/images/KKomrade.png',

],

[

'id' => 7,

'name' => '5Head',

'description' => 'Intellectual.',

'image' => '@/assets/images/5Head.png',

]

];

return $emotes;

}

}

**UserController.php**

<?php

namespace app\modules\v1\controllers;

use app\modules\v1\models\User;

use yii\rest\Controller;

class UserController extends ApiController {

public function actionLogin() {

// login stuff

}

public function actionRegister() {

// register stuff

}

public function actionAll() {

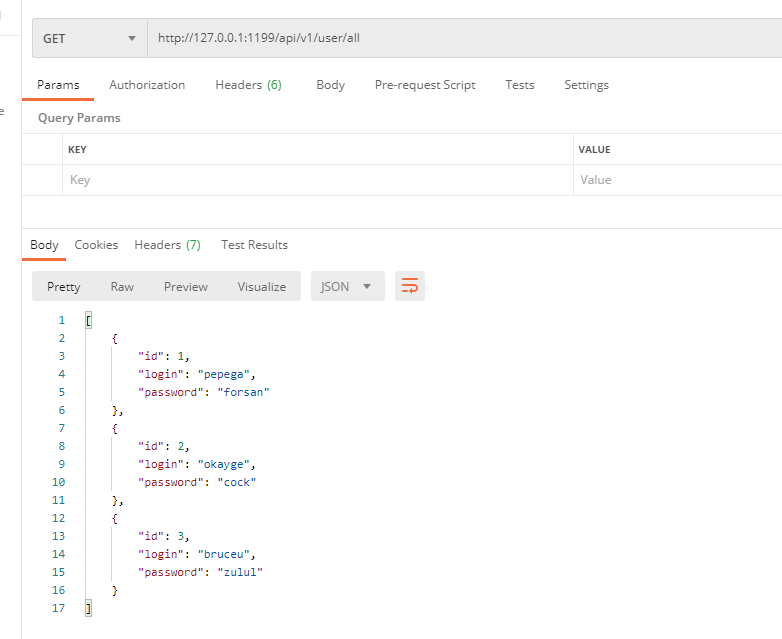
$user = new User();

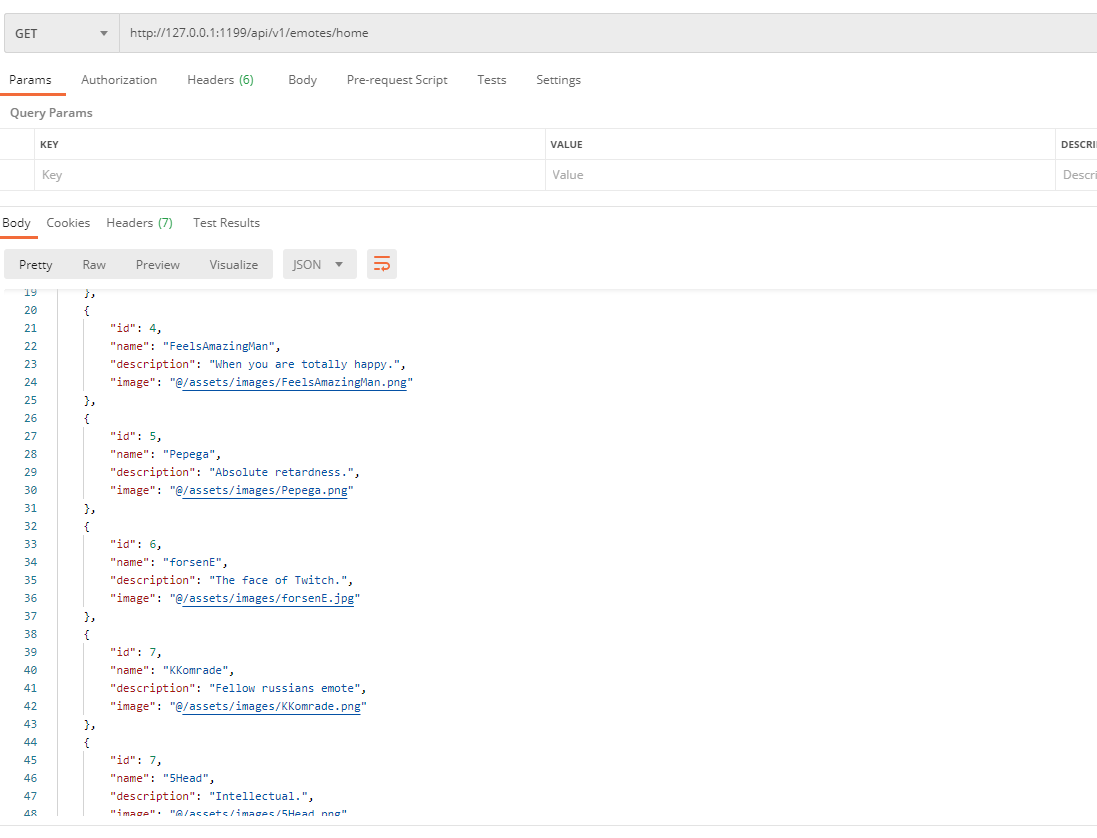
return $user -> getUsers();

}

}

Дале были проверены запросы в Postman:





**Вывод:** В результате выполнения работы были изучены основы backend-разработки web-приложений и скрипты на языке PHP. Также были рассмотрены основы работы с docker и фреймворком Yii2 а также приобретены навыки работы с API в приложении Postman.