МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РФ
ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ БЮДЖЕТНОЕ
ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ
«БЕЛГОРОДСКИЙ ГОСУДАРСТВЕННЫЙ ТЕХНОЛОГИЧЕСКИЙ УНИВЕРСИТЕТ им. В. Г. ШУХОВА»
(БГТУ им. В.Г. Шухова)

Кафедра программного обеспечения вычислительной техники и автоматизированных систем

Лабораторная работа № 3
дисциплина: Технологии web-программирования
тема: «Серверное программирование. Yii2»

Выполнил: ст. гр. ПВ-42
Богров Дмитрий Алексеевич
Проверил: Картамышев С. В.

Белгород 2020

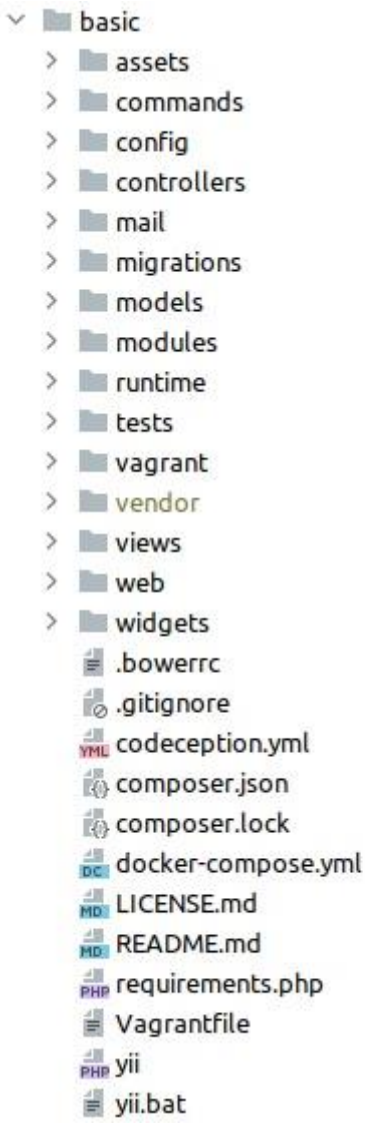
Цель: познакомиться с основами backend разработки web-приложений. Научится писать скрипты на языке PHP. Познакомиться с основами работы docker. Познакомиться с фреймворком Yii2 и научиться разворачивать проект, производить его настройку. Научится работать с API в приложении Postman.

Задание к 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.

Ход работы

Базовое приложение Yii2 разворачивается с помощью php-композера командой `composer create-project --prefer-dist yiisoft/yii2-app-basic basic`. В результате выполнения команды были получены следующие директории:



Перед запуском сервера необходимо настроить 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'

  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!
```

Для запуска сервера используется команда **docker-compose up -d**.

После проделанных операций можно перейти к добавлению контроллеров со статическими данными (перед этим создав контроллер для разрешения cors-запросов).

ApiController.php:

```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
            ]
        ];
    }
}
```

PageController.php:

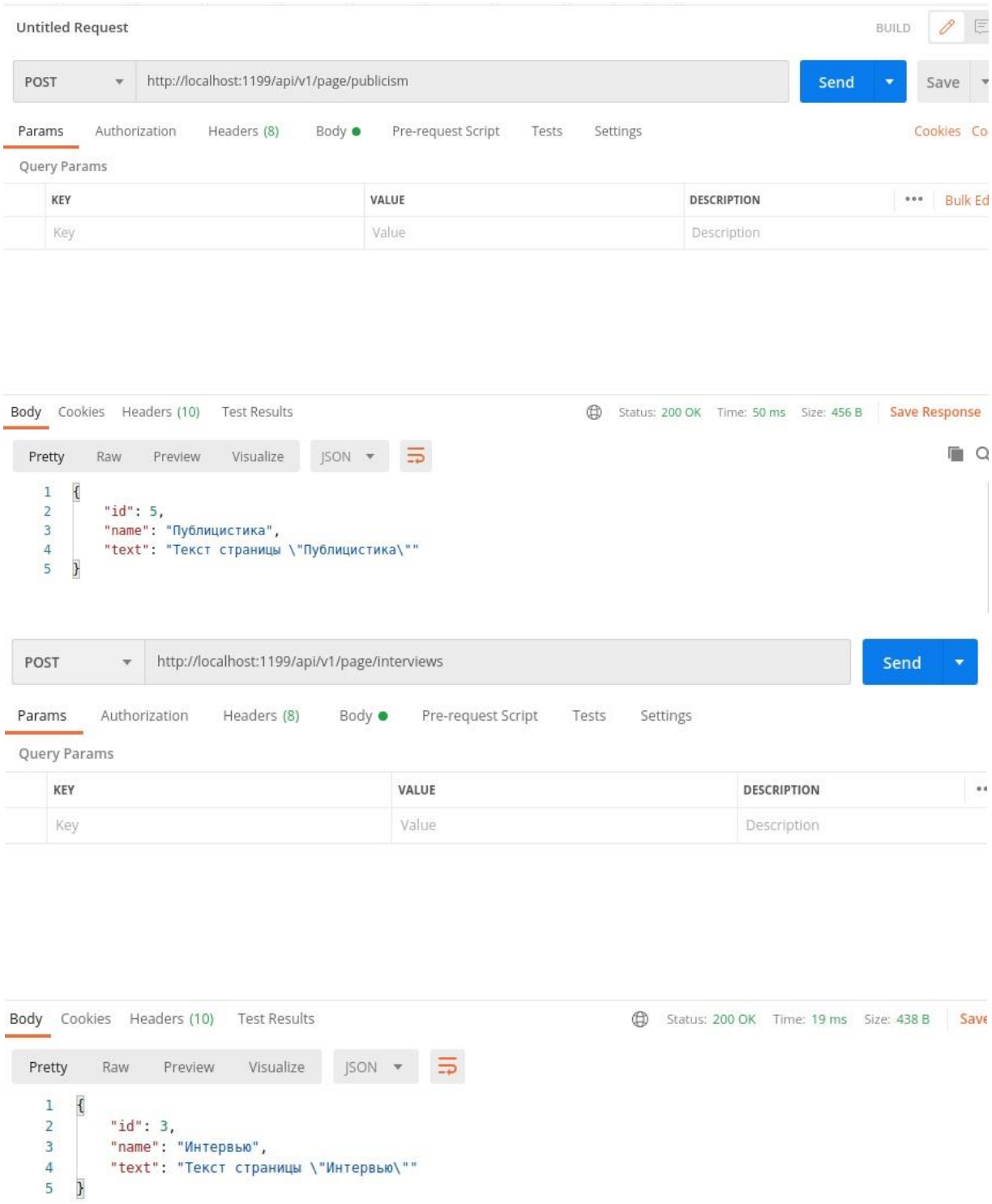
```
<?php

namespace app\modules\v1\controllers;


class PageController extends ApiController
{
    public function actionPublicism ()
    {
        return [
            'id' => 5,
            'name' => 'Публицистика',
            'text' => 'Текст страницы "Публицистика"'
        ];
    }

    public function actionInterviews ()
    {
        return [
            'id' => 3,
            'name' => 'Интервью',
            'text' => 'Текст страницы "Интервью"'
        ];
    }
}
```

Скриншоты работы контроллеров в Postman:



Вывод: при выполнении работы мы ознакомились с основами backend разработки web-приложений, научились писать скрипты на языке PHP. Познакомились с основами работы docker. Познакомились с фреймворком Yii2 и научились разворачивать проект, производить его настройку. Научились работать с API в приложении Postman.