МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ ФЕДЕРАЦИИ 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: студент группы ВТ-41

Шкорба В. С.

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 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**

В ходе работы было развёрнуто базовое приложение Yii2 App Basic с помощью команды: “composer create-project --prefer-dist yiisoft/yii2-app-basic basic”. Затем это приложение было настроено для работы и дополнено результатами предыдущих лабораторных работ. Константные данный были перенесены в бэкенд сайта и вызываются посредством REST API.

Для настройки конфигурации изменили файл docker-compose.yaml:

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:

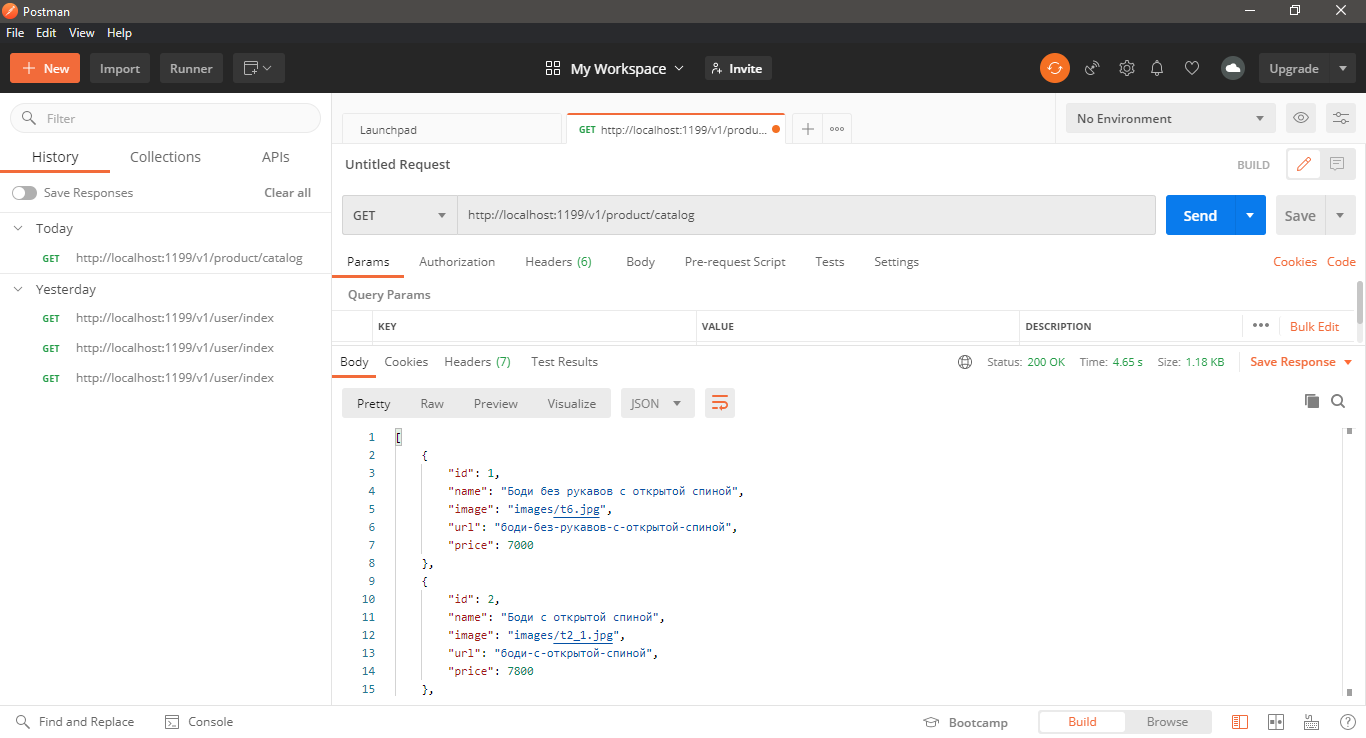
      - MYSQL\_ROOT\_PASSWORD=verysecret

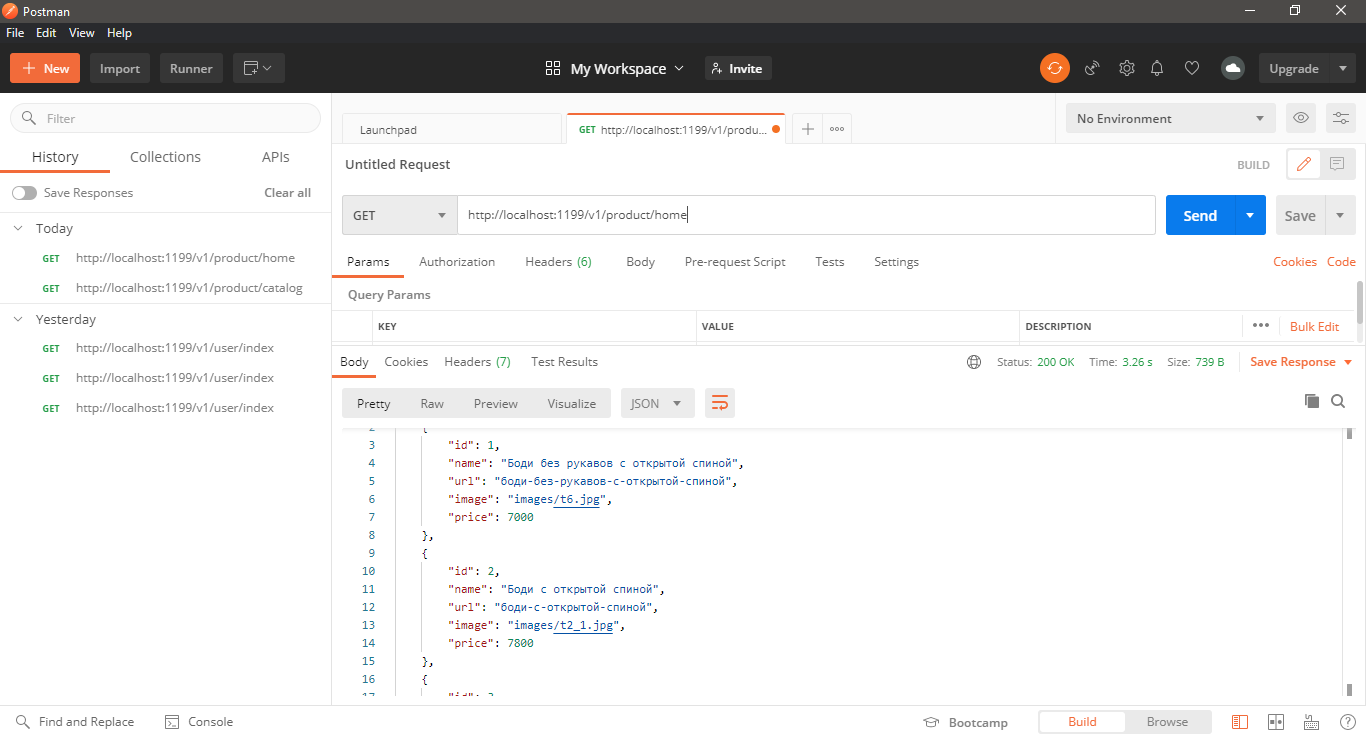
      - MYSQL\_DATABASE=amour

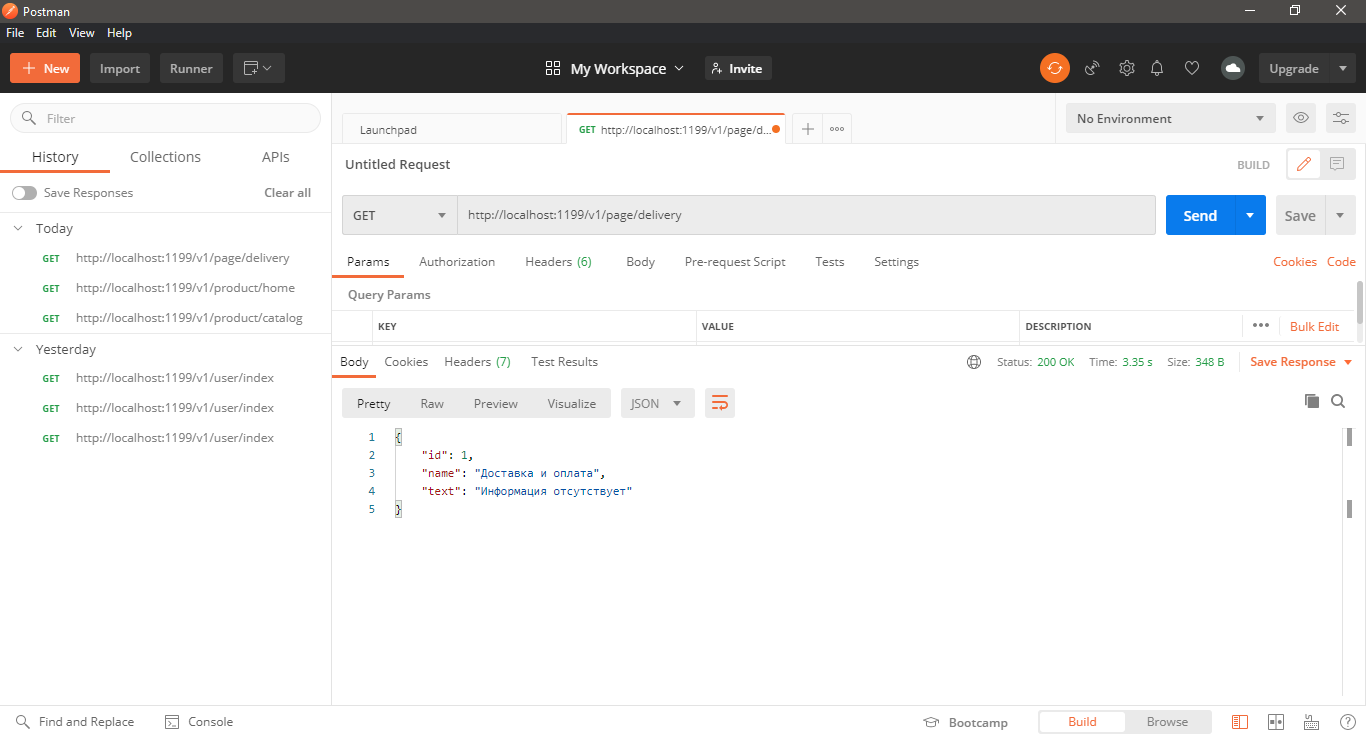
      - MYSQL\_USER=amour

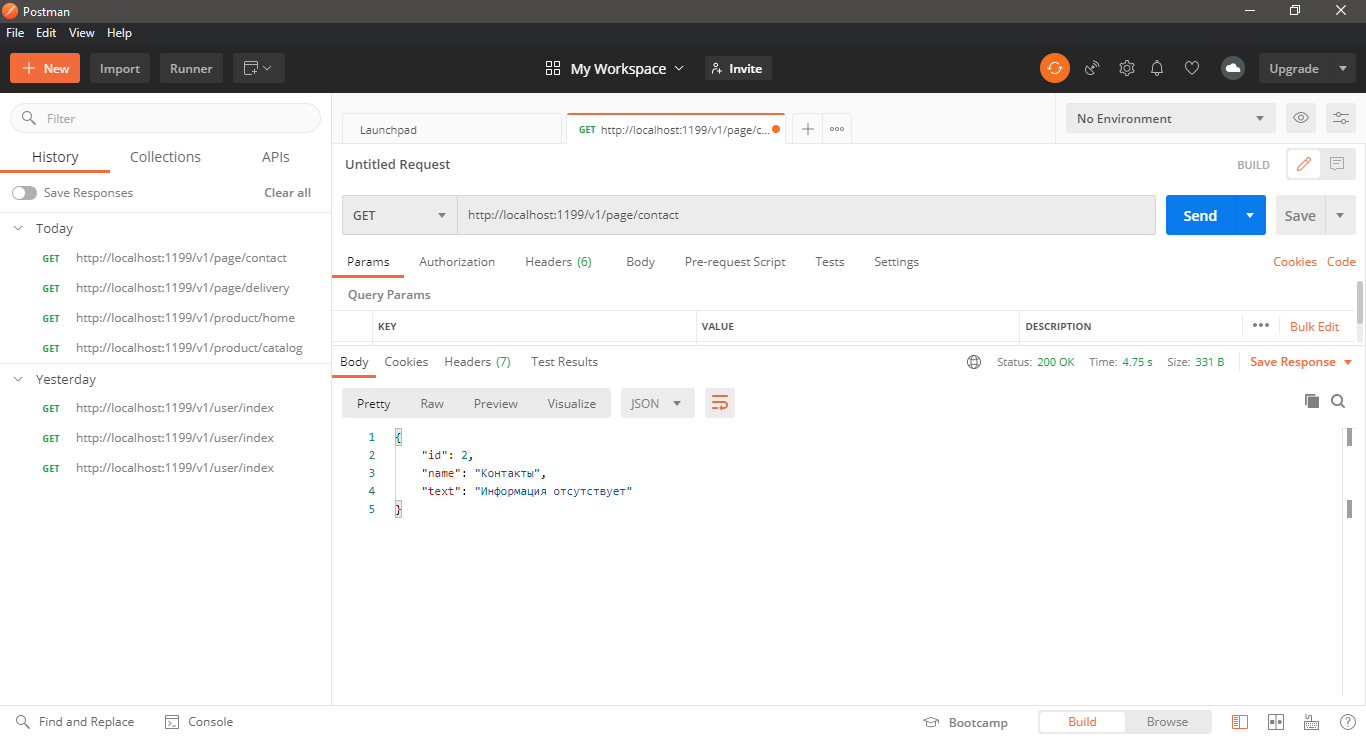
      - MYSQL\_PASSWORD=amour<3

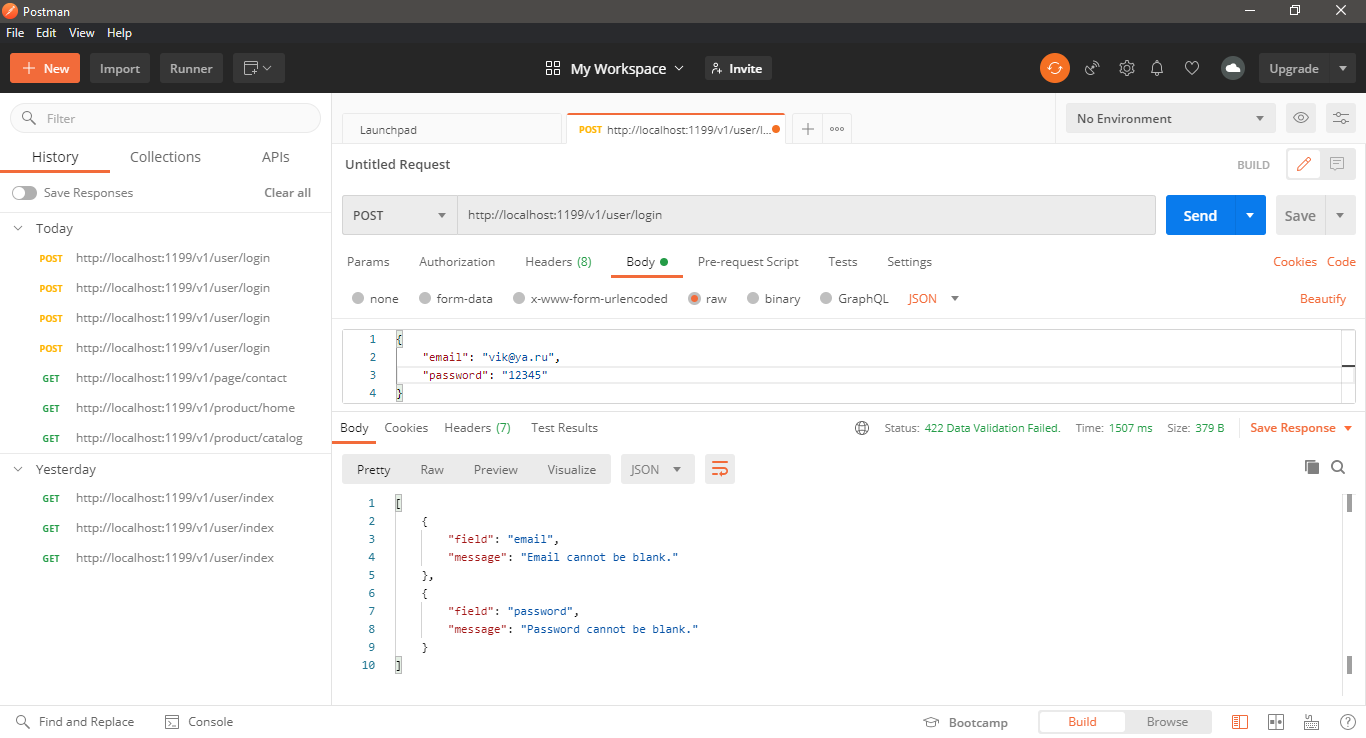
Результаты работы запросов в Postman:











Приложение 1. Основные исходные коды.

«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

            ]

        ];

    }

}

«PageController.php»

<?php

namespace app\modules\v1\controllers;

class PageController extends ApiController

{

    public function actionDelivery()

    {

        return [

            'id' => 1,

            'name' => 'Доставка и оплата',

            'text' => 'Информация отсутствует'

        ];

    }

    public function actionContact()

    {

        return [

            'id' => 2,

            'name' => 'Контакты',

            'text' => 'Информация отсутствует'

        ];

    }

}

«UserController.php»

<?php

namespace app\modules\v1\controllers;

use app\modules\v1\models\LoginForm;

use yii\rest\Controller;

class UserController extends ApiController

{

    public function actionLogin()

    {

        $form = new LoginForm();

        $form->load(\Yii::$app->request->getBodyParams(), '');

        if ($form->validate()) {

            $form->login();

            return $form->getUser();

        }

        return $form;

    }

}

«LoginForm.php»

<?php

namespace app\modules\v1\models;

use yii\base\Model;

class LoginForm extends Model

{

    public $email;

    public $password;

    public function rules()

    {

        return [

            [['email', 'password'], 'required'],

            [['email'], 'email', 'message' => 'Некорректный Email.'],

            [['email', 'password'], 'string', 'max' => 64, 'min' => 3]

        ];

    }

    public function login()

    {

        // TODO: сделать авторизацию

    }

    public function getUser()

    {

        return [

            'id' => 1,

            'name' => 'Виктория',

            'email' => $this->email,

            'accessToken' => md5(microtime(true))

        ];

    }

}

«Signin.vue»

<template>

  <div class="container sign-in">

    <form class="form-signin">

      <h1 class="mb-4 font-weight-normal text-center">Авторизация</h1>

      <form @submit="signIn">

        <label for="email" class="sr-only">Email address</label>

        <input type="email" id="email" class="form-control" v-model="email">

        <div class="error">{{ errors.email }}</div>

        <label for="password" class="sr-only">Password</label>

        <input

          type="password"

          id="password"

          class="form-control mb-3"

          v-model="password">

        <div class="error">{{ errors.password }}</div>

        <div class="checkbox text-center mb-2">

          <label>

            <input type="checkbox" value="remember-me"> Запомнить меня

          </label>

        </div>

        <button class="btn btn-lg btn-secondary btn-block" type="submit">

          Войти

        </button>

      </form>

      <p class="reg mt-3 text-muted">

        <a href="#">Забыли пароль?</a> &middot;

        <router-link v-bind:to="{ name: 'CheckIn' }">Регистрация</router-link>

      </p>

    </form>

  </div>

</template>

<script>

import User from "@/components/user/user";

import router from "@/router";

import Axios from "axios";

export default {

  name: "SignIn",

  data() {

    return {

      email: '',

      password: '',

      errors: {}

    }

  },

  methods: {

    signIn(e) {

      if (this.validate()) {

        const instance = Axios.create({

          baseURL: 'http://localhost:1199/v1',

        });

        instance.post('/user/login', {

            email: this.email,

            password: this.password

          }).then((response) => {

            User.login(response.data);

            router.push({name: 'Home'});

          })

          .catch((errors) => {

            console.log(errors.response.data)

          })

      }

      e.preventDefault();

    },

    validate() {

      this.errors = {};

      if (this.email.trim().length === 0) {

        this.errors.email = "Заполните Email.";

      }

      if (this.password.trim().length === 0) {

        this.errors.password = "Заполните Пароль.";

      }

      return Object.keys(this.errors).length === 0;

    },

  },

};

</script>