**Практическая работа по МДК02.02**

I. В зависимости от **варианта** (согласно номеру в электронном журнале) требуется интегрировать на свой сайт, созданный посредством фреймворка Bottle, **страницу**:

4. Отзывов

II. Подключить систему контроля версий **Git**, фиксировать основные изменения интеграции, передать их в удалённый репозиторий.

III. Написать не менее 2 unit-тестов для проверки ввода данных (даты или телефона). Выполнить их запуск из Обозревателя тестов, обработку всех исключений, отладку модулей и всей интеграции.

IV. Оформить отчёт (отдельный документ), в котором отобразить **ТЗ**, новый вариант **UML**-диаграммы, **код** с комментариями, **скриншоты** главной (панель навигации должна дополнена) и новой страниц сайта, результатов **тестирования**, **отладки** и работы **системы контроля версий**.

**ТЗ**

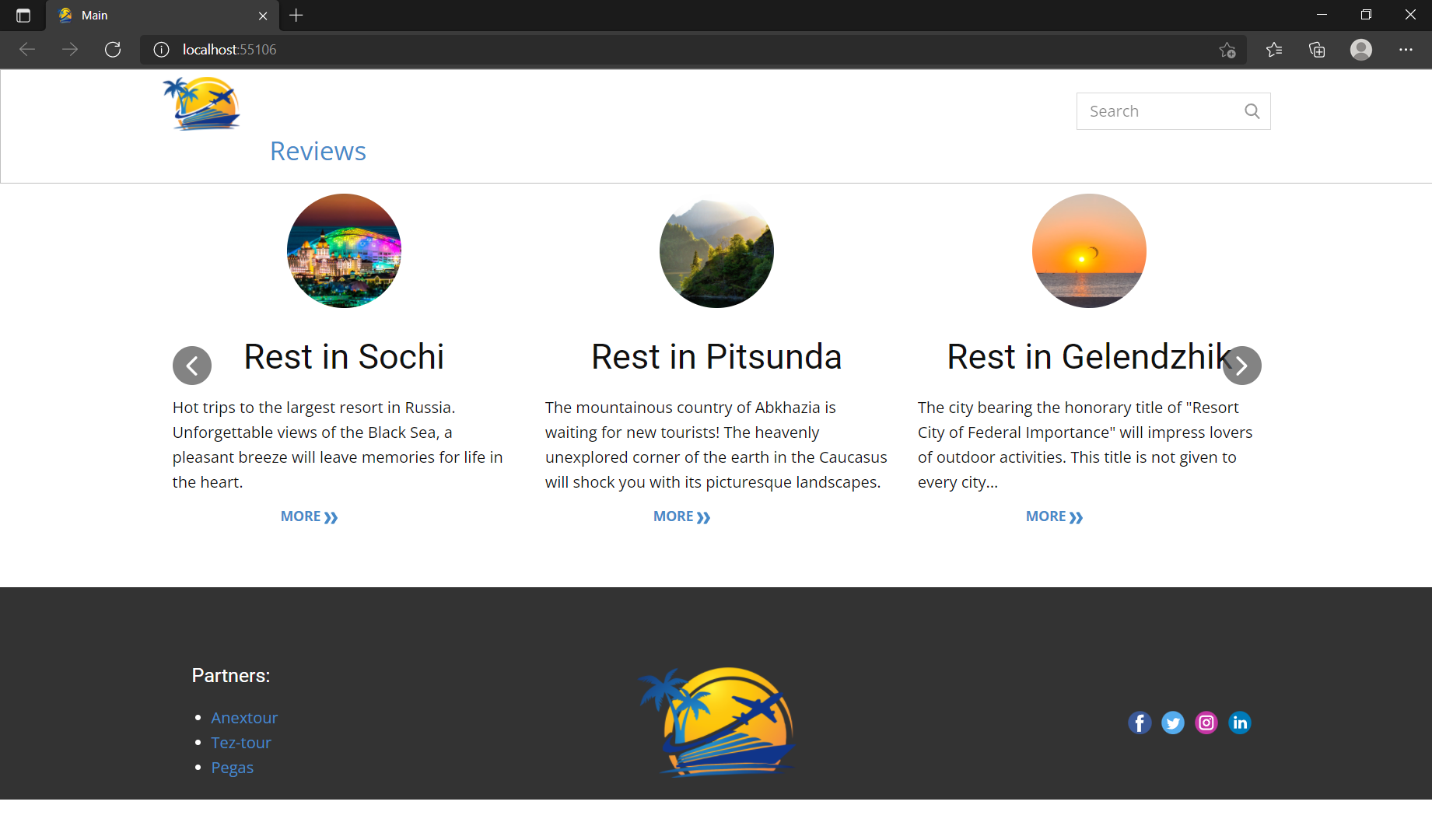
Необходимо реализовать страницу отзывов на туристическом сайте.

На странице должна быть форма записи новых отзывов, а также вывод уже имеющихся из отдельного файла.

При вводе нового отзыва должны учитываться отзыв, e-mail пользователя, а также номер телефона. Для каждой записи необходимо соотносить со временем её создания.

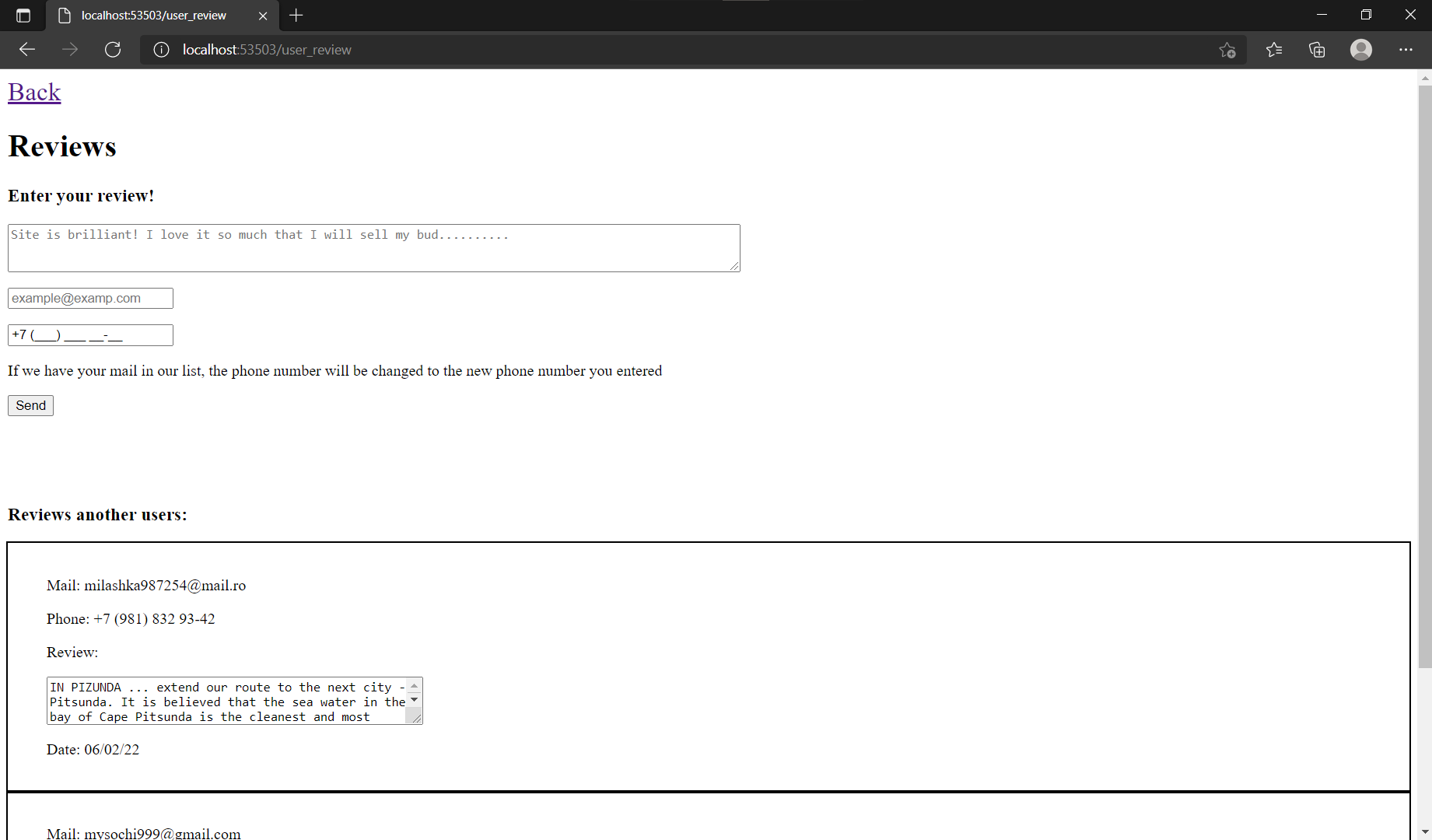
Доступ к странице с отзывами должен осуществляться через навигационную панель

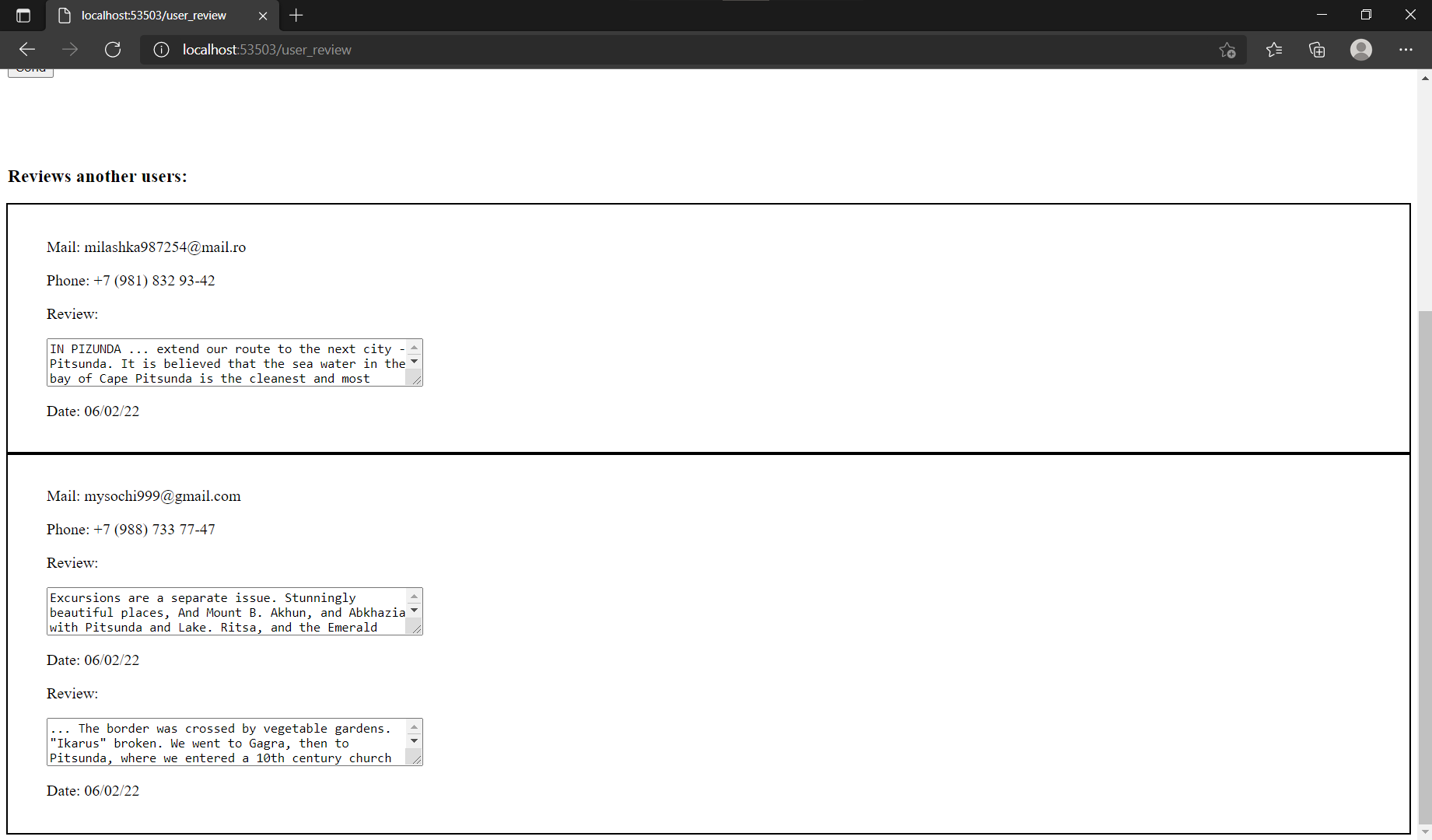
**Скриншоты страниц**



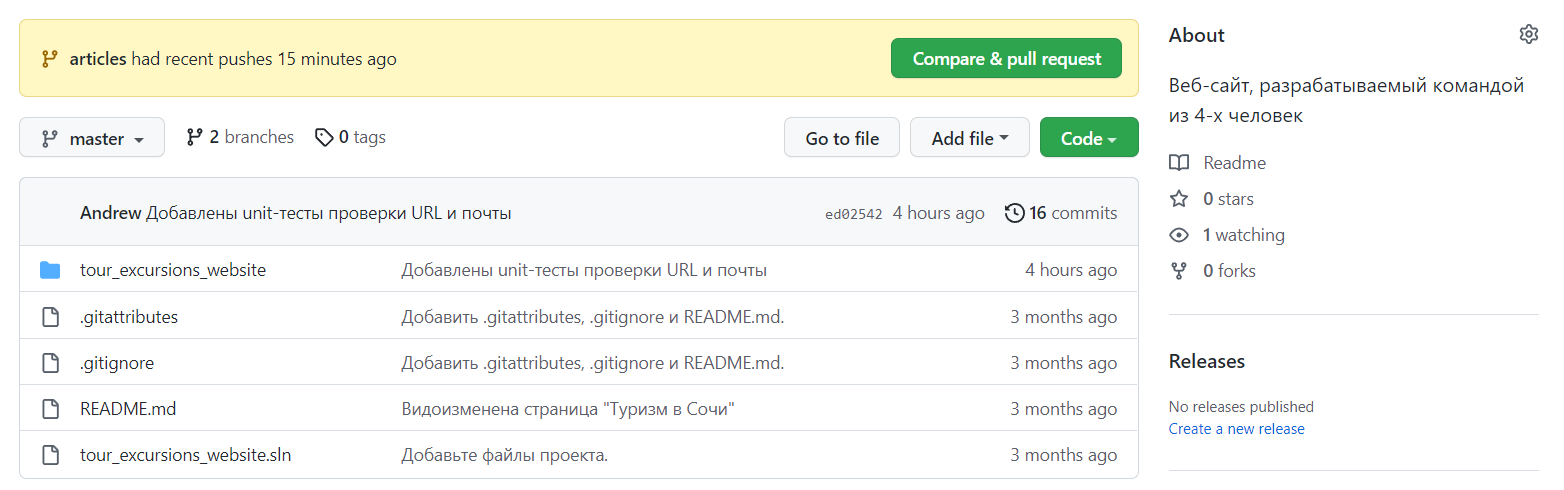
В связи с тем, что при работе с фреймворком возникла проблема с кодировкой, сайт решено было перевести на английский

Скриншот созданной страницы c отзывами:

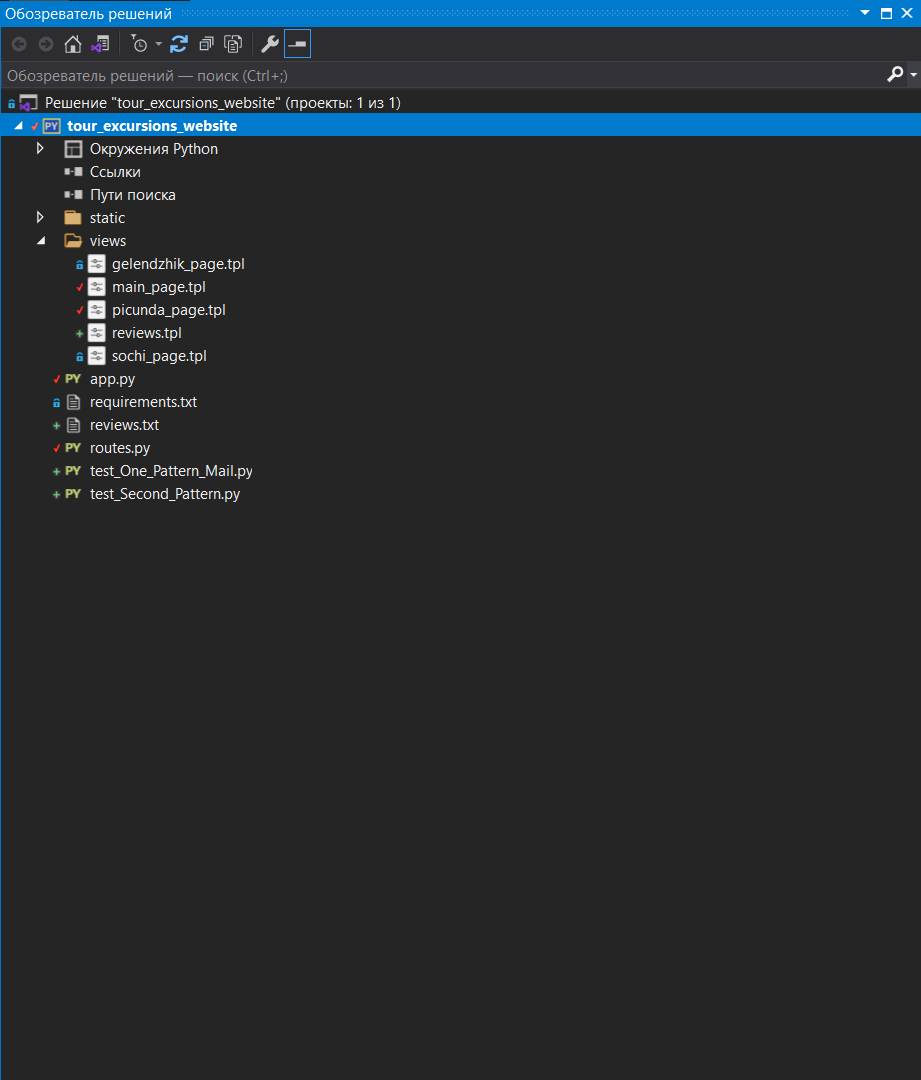




**Скриншоты обозревателя решений VS и GitHub**

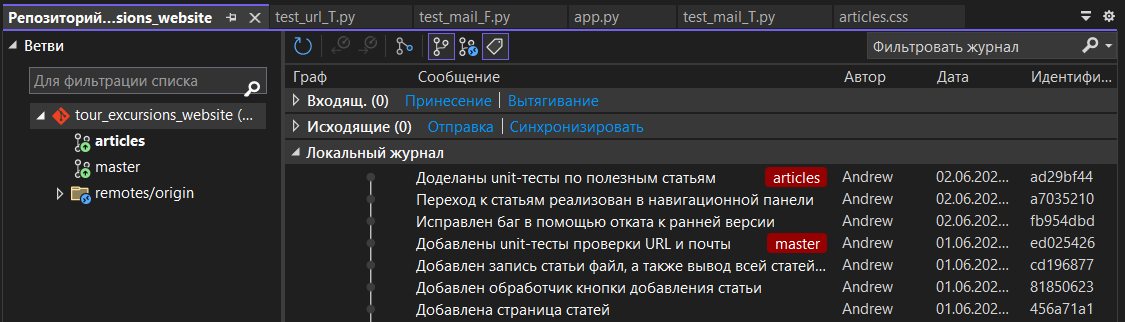


Обозреватель GitHub

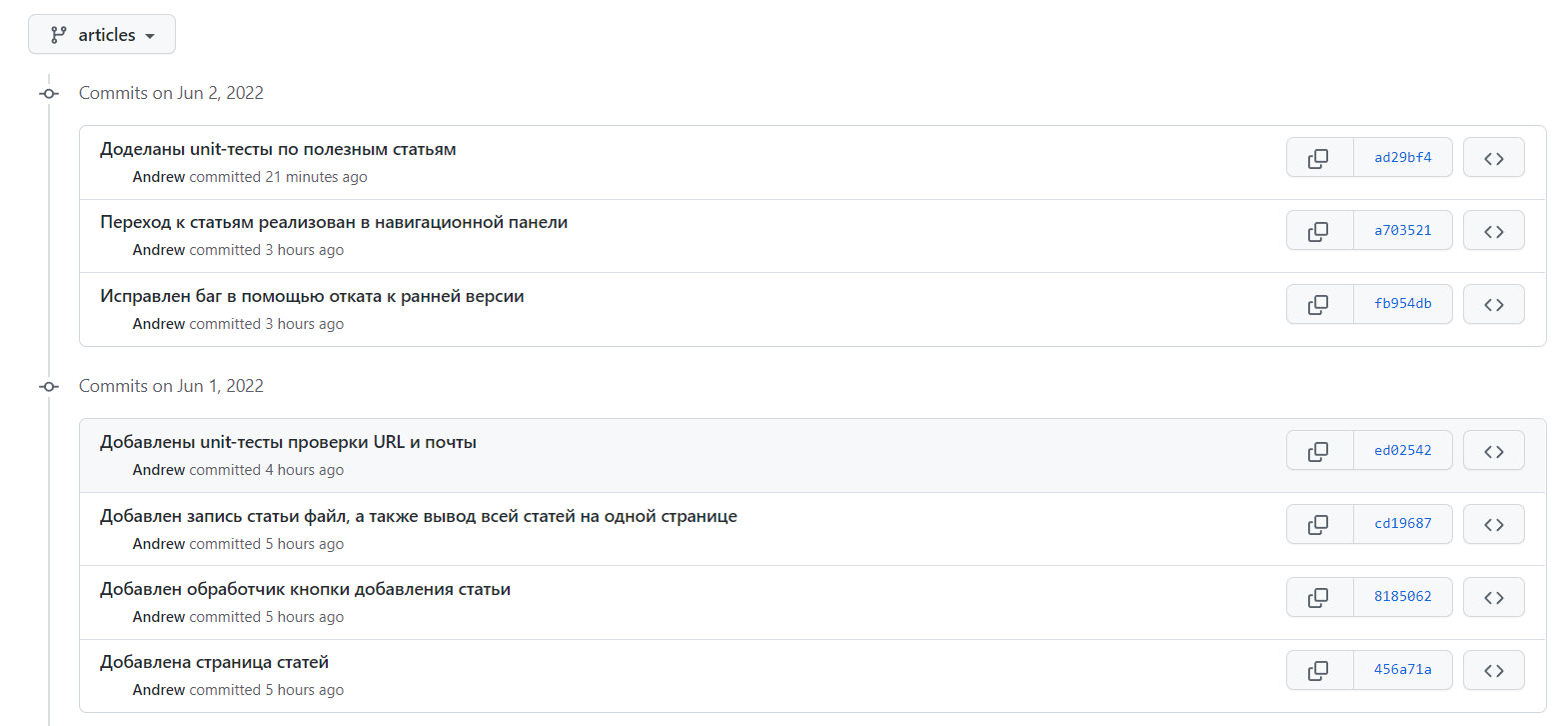


Обозреватель решений Visual Studio

**Скриншоты коммитов в VS и GitHub**



История коммитов в VS



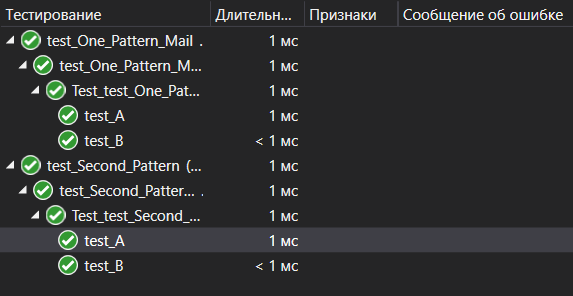
История коммитов в GitHub

**UML-диаграммы**

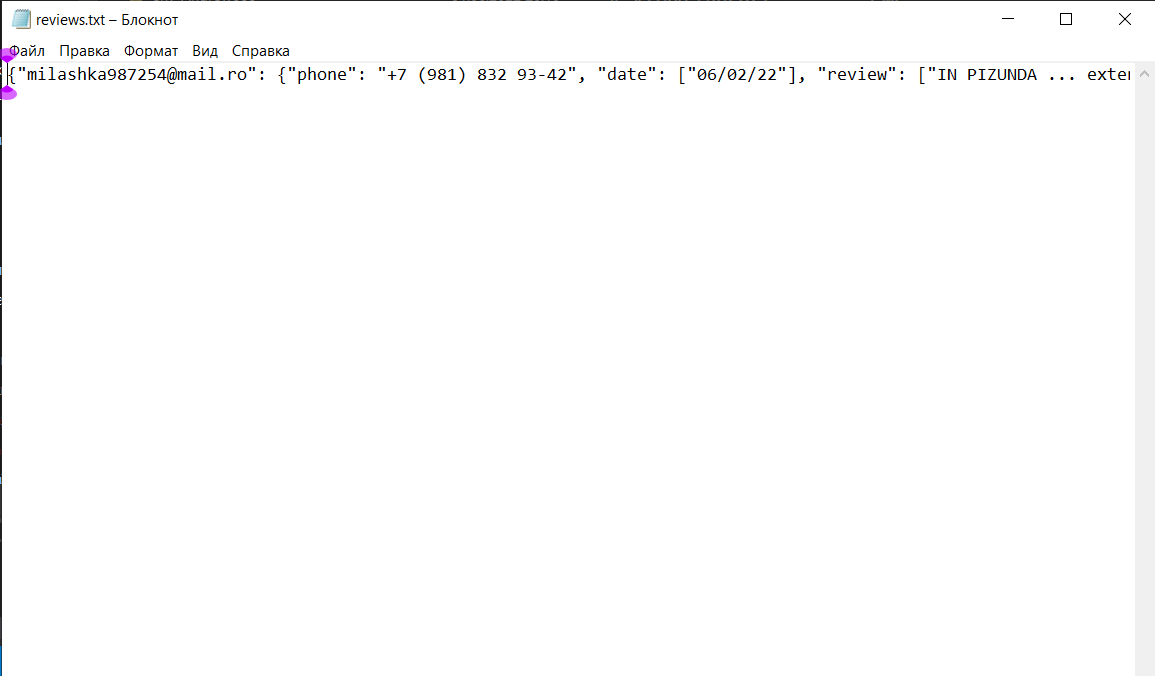


UML-диаграмма компонентов

**Скриншот обозревателя тестов и записи в файл**



Выполненные 4 unit-теста



Файл для записи статей в директории static/reviews.txt

**Листинг модуля внесения нового отзыва**

from bottle import route, view, post, template, request

import bottle

from datetime import datetime

import json

import os

import sys

@post('/user\_review')

@view('reviews')

def review\_form():

mail = request.forms.get('MAIL')

phone = request.forms.get('PHONE')

review = request.forms.get('REVIEW')

date = datetime.now()

if (os.path.exists('reviews.txt')):

with open('reviews.txt',encoding='utf8') as json\_file:

reviews = json.load(json\_file)

else:

reviews = {}

print(reviews)

if (mail in reviews.keys()):

reviews[mail]['phone'] = phone

reviews[mail]['review'].append(review)

reviews[mail]['date'].append(date.strftime("%x"))

else:

reviews[mail] = {'phone': phone, 'date': [date.strftime("%x")], 'review': [review]}

print(reviews)

with open('reviews.txt', 'w', encoding='utf8') as outfile:

json.dump(reviews, outfile)

return template('reviews')

**Листинг шаблона страницы с полезными статьями**

<!DOCTYPE html>

<header\_char>

<head>

<a href="/" style="font-size: 20pt">Back</a>

<h1>Reviews</h1>

<meta charset="utf-8">

<link rel="stylesheet" type="text/css" href="/static/content/site.css" />

</head>

<body id = back1>

<form id="form\_user\_review" action = "/user\_review" method = "post">

<h3> Enter your review! </h3>

<p><textarea input type="text" rows="3" cols="100" name="REVIEW" placeholder="Site is brilliant! I love it so much that I will sell my bud.........." id="review" maxlength = "400"></textarea></p>

<p><input type="text" name="MAIL" placeholder="example@examp.com" id="mail"

pattern="^[a-zA-Z0-9\_.+-]+@[a-z]+.[a-z]{2,3}$" title="Enter mail in right format - example@mail.com"></p>

<p><input type="tel" name = "PHONE" id="PHONE" value = "+7 (\_\_\_) \_\_\_ \_\_-\_\_" placeholder="Enter mobile phone number" id="phone"

pattern="^\+7(\s+)?\(?[0-9]{3}\)?(\s+)?[0-9]{3}?(\s+)?[0-9]{2}-?[0-9]{2}$" requered title="Enter mobile phone number in right format - +# (###) ### ##-##"></p>

<p>If we have your mail in our list, the phone number will be changed to the new phone number you entered</p>

<p><input type="submit" value="Send" class="btn btn-default" id="btn" onclick = "onClickReview()"></p>

</form>

<br /><br /><br />

<%import json%>

<%import os%>

<%import sys%>

<%if (os.path.exists('reviews.txt')):%>

<%with open('reviews.txt',encoding='utf8') as json\_file:%>

<%reviews = json.load(json\_file)%>

<%print(reviews)%>

<%if len(reviews) > 0:%>

<%c=0%>

<h3> Reviews another users:</h3>

<%for mail in reviews: %>

<%c+=1%>

<div style="outline: 2px solid #000">

<form id="form\_reviews\_{{c}}">

<br />

<ol>

<p>Mail: {{mail}}</p>

<p>Phone: {{reviews[mail]['phone']}}</p>

</ol>

<ol>

<%for j in range(len(reviews[mail]['date'])):%>

<p>Review:</p>

<p><textarea readonly="readonly" input type="text" rows="3" cols="50">{{reviews[mail]['review'][j]}}</textarea></p>

<p>Date: {{reviews[mail]['date'][j]}}</p>

<%end%>

</ol>

</form>

<br />

</div>

<%end%>

<%end%>

<%end%>

<%end%>

<script>

window.onload = function() {

MaskedInput({

elm: document.getElementById('PHONE'), // select only by id

format: '+7 (\_\_\_) \_\_\_ \_\_-\_\_',

separator: '+7 ()-'

});

};

(function(a){a.MaskedInput=function(f){if(!f||!f.elm||!f.format){return null}if(!(this instanceof a.MaskedInput)){return new a.MaskedInput(f)}var o=this,d=f.elm,s=f.format,i=f.allowed||"0123456789",h=f.allowedfx||function(){return true},p=f.separator||"/:-",n=f.typeon||"\_YMDhms",c=f.onbadkey||function(){},q=f.onfilled||function(){},w=f.badkeywait||0,A=f.hasOwnProperty("preserve")?!!f.preserve:true,l=true,y=false,t=s,j=(function(){if(window.addEventListener){return function(E,C,D,B){E.addEventListener(C,D,(B===undefined)?false:B)}}if(window.attachEvent){return function(D,B,C){D.attachEvent("on"+B,C)}}return function(D,B,C){D["on"+B]=C}}()),u=function(){for(var B=d.value.length-1;B>=0;B--){for(var D=0,C=n.length;D<C;D++){if(d.value[B]===n[D]){return false}}}return true},x=function(C){try{C.focus();if(C.selectionStart>=0){return C.selectionStart}if(document.selection){var B=document.selection.createRange();return -B.moveStart("character",-C.value.length)}return -1}catch(D){return -1}},b=function(C,E){try{if(C.selectionStart){C.focus();C.setSelectionRange(E,E)}else{if(C.createTextRange){var B=C.createTextRange();B.move("character",E);B.select()}}}catch(D){return false}return true},m=function(D){D=D||window.event;var C="",E=D.which,B=D.type;if(E===undefined||E===null){E=D.keyCode}if(E===undefined||E===null){return""}switch(E){case 8:C="bksp";break;case 46:C=(B==="keydown")?"del":".";break;case 16:C="shift";break;case 0:case 9:case 13:C="etc";break;case 37:case 38:case 39:case 40:C=(!D.shiftKey&&(D.charCode!==39&&D.charCode!==undefined))?"etc":String.fromCharCode(E);break;default:C=String.fromCharCode(E);break}return C},v=function(B,C){if(B.preventDefault){B.preventDefault()}B.returnValue=C||false},k=function(B){var D=x(d),F=d.value,E="",C=true;switch(C){case (i.indexOf(B)!==-1):D=D+1;if(D>s.length){return false}while(p.indexOf(F.charAt(D-1))!==-1&&D<=s.length){D=D+1}if(!h(B,D)){c(B);return false}E=F.substr(0,D-1)+B+F.substr(D);if(i.indexOf(F.charAt(D))===-1&&n.indexOf(F.charAt(D))===-1){D=D+1}break;case (B==="bksp"):D=D-1;if(D<0){return false}while(i.indexOf(F.charAt(D))===-1&&n.indexOf(F.charAt(D))===-1&&D>1){D=D-1}E=F.substr(0,D)+s.substr(D,1)+F.substr(D+1);break;case (B==="del"):if(D>=F.length){return false}while(p.indexOf(F.charAt(D))!==-1&&F.charAt(D)!==""){D=D+1}E=F.substr(0,D)+s.substr(D,1)+F.substr(D+1);D=D+1;break;case (B==="etc"):return true;default:return false}d.value="";d.value=E;b(d,D);return false},g=function(B){if(i.indexOf(B)===-1&&B!=="bksp"&&B!=="del"&&B!=="etc"){var C=x(d);y=true;c(B);setTimeout(function(){y=false;b(d,C)},w);return false}return true},z=function(C){if(!l){return true}C=C||event;if(y){v(C);return false}var B=m(C);if((C.metaKey||C.ctrlKey)&&(B==="X"||B==="V")){v(C);return false}if(C.metaKey||C.ctrlKey){return true}if(d.value===""){d.value=s;b(d,0)}if(B==="bksp"||B==="del"){k(B);v(C);return false}return true},e=function(C){if(!l){return true}C=C||event;if(y){v(C);return false}var B=m(C);if(B==="etc"||C.metaKey||C.ctrlKey||C.altKey){return true}if(B!=="bksp"&&B!=="del"&&B!=="shift"){if(!g(B)){v(C);return false}if(k(B)){if(u()){q()}v(C,true);return true}if(u()){q()}v(C);return false}return false},r=function(){if(!d.tagName||(d.tagName.toUpperCase()!=="INPUT"&&d.tagName.toUpperCase()!=="TEXTAREA")){return null}if(!A||d.value===""){d.value=s}j(d,"keydown",function(B){z(B)});j(d,"keypress",function(B){e(B)});j(d,"focus",function(){t=d.value});j(d,"blur",function(){if(d.value!==t&&d.onchange){d.onchange()}});return o};o.resetField=function(){d.value=s};o.setAllowed=function(B){i=B;o.resetField()};o.setFormat=function(B){s=B;o.resetField()};o.setSeparator=function(B){p=B;o.resetField()};o.setTypeon=function(B){n=B;o.resetField()};o.setEnabled=function(B){l=B};return r()}}(window));

</script>

</body>

</header\_char>

**Листинг unit-тестов**

Листинг unit-теста на проверку валидности почты с набором некорректных данных и на проверку валидности почты с корректным набором данных:

import unittest

import re

class Test\_test\_One\_Pattern\_Mail(unittest.TestCase):

def test\_A(self):

pattern = "^[a-zA-Z0-9\_.+-]+@[a-z]{3,6}.[a-z]{2,3}$"

wrongMails = [""," s@gmail.com","sseq21","agadga@","adgda@kkkkkkk.com","adghfgjfds@dfgd.ddddddddd"]

for mail in wrongMails:

self.assertFalse(re.match(pattern,mail))

def test\_B(self):

pattern = "^[a-zA-Z0-9\_.+-]+@[a-z]{3,6}.[a-z]{2,3}$"

rightMails = ["9876@gmail.com","andrew@mail.com","a@gmail.com","andy@mail.ru"]

for mail in rightMails:

self.assertTrue(re.match(pattern,mail))

if \_\_name\_\_ == '\_\_main\_\_':

unittest.main()

Листинг unit-теста на проверку паттерна телефона с корректным и некорректным набором данных:

import unittest

import re

class Test\_test\_Second\_Pattern(unittest.TestCase):

def test\_A(self):

pattern = "^\+7(\s+)?\(?[0-9]{3}\)?(\s+)?[0-9]{3}?(\s+)?[0-9]{2}-?[0-9]{2}$"

wrongNumbers = ["8998765677"," ","+8 (986) 765 87-42","+7 (987) 765 87-443"]

for number in wrongNumbers:

self.assertFalse(re.match(pattern,number))

def test\_B(self):

pattern = "^\+7(\s+)?\(?[0-9]{3}\)?(\s+)?[0-9]{3}?(\s+)?[0-9]{2}-?[0-9]{2}$"

rightNumbers = ["+7 (965) 234 34-21", "+798676565-45","+7(877)876 87-76"]

for number in rightNumbers:

self.assertTrue(re.match(pattern,number))

if \_\_name\_\_ == '\_\_main\_\_':

unittest.main()