**ОТЧЕТ ПО УЧЕБНОЙ ПРАКТИКЕ**

**Выполнила:**

Кучминская А. С., 2 курс 7 группа

**Руководитель:**

Зенько Т. А.

Минск, 2021

**ОГЛАВЛЕНИЕ**

***4*** *семестр*

1. Лабораторная работа №1 3
2. Лабораторная работа №2 5
3. Лабораторная работа №3 9
4. Лабораторная работа №4 20
5. Лабораторная работа №5 25
6. Лабораторная работа №7 29
7. Использованная литература 33

***4*** *семестр*

**ЛАБОРАТОРНАЯ РАБОТА №1**

*Работа с Git/GitHub*

**УСЛОВИЕ:**

Установка и настройка. Создание нового локального репозитория и основы работы с ним. Создание профиля на GitHub и удаленного репозитория. Синхронизация локального и удаленного репозиториев. Копирование существующего удаленного репозитория и работа с ним. Проверка результатов работы с удаленным репозиторием.

**РЕАЛИЗАЦИЯ:**

Установка Git, создание иерархии папок и файлов. Создание проекта. Работа через командную строку

*Git init --bare* – создание пустого репозитория

*Git config –* инициализация пользователя

*Git add* – добавление файлов и папок для слежки

*Git status* – проверка статуса файлов

Создание файл .gitignore в корне репозитория.

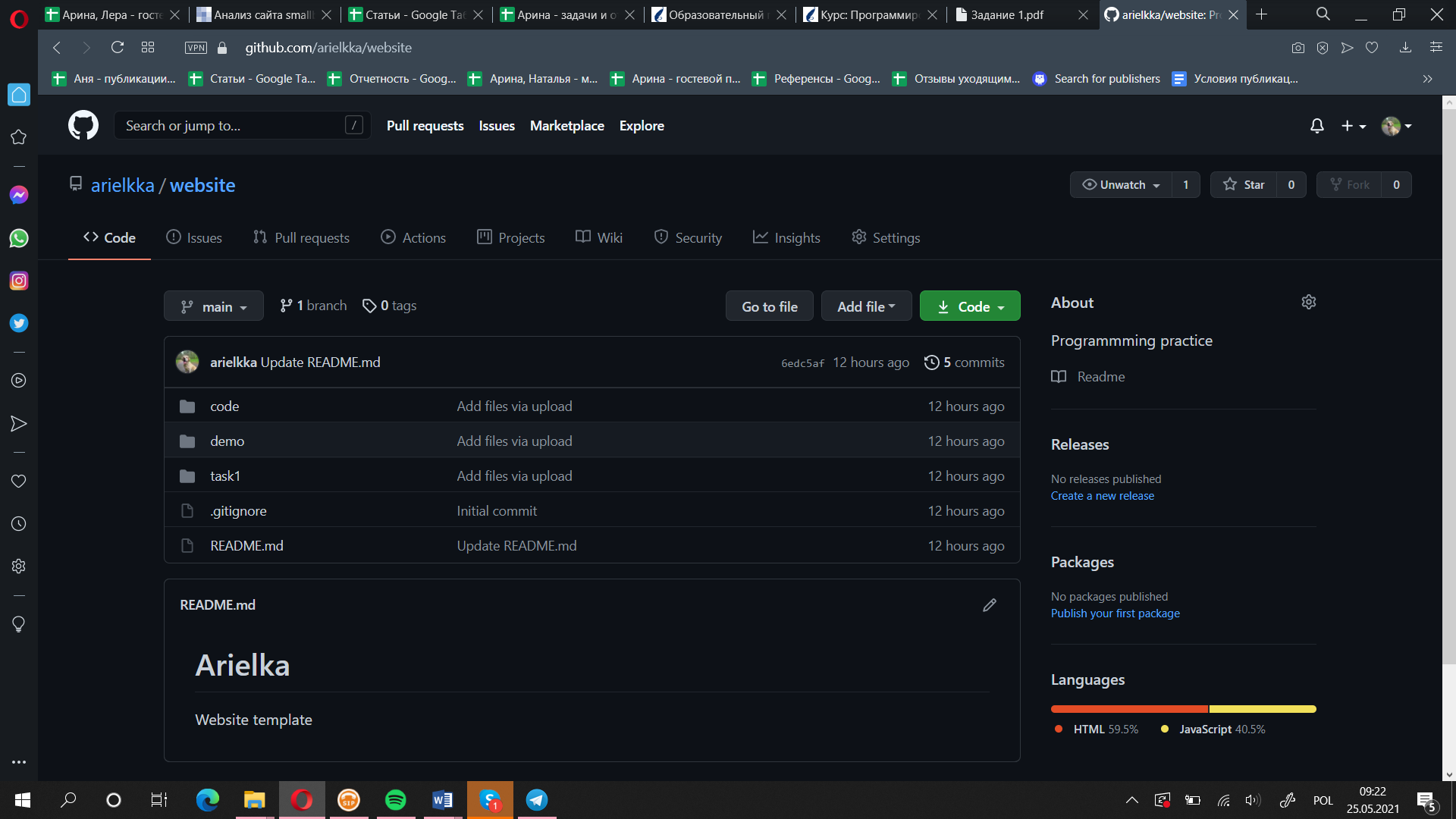
*Git commit -m “add .gitignore”* - фиксирование изменения в локальном репозитории.

*Git remote add origin* - привязка удаленного репозитория к локальному

*Git clone* - клонируем наш репозиторий

*Git pull* - забираем изменения с удаленного репозитория

**РЕЗУЛЬТАТ ВЫПОЛНЕНИЯ:**

****

**ЛАБОРАТОРНАЯ РАБОТА №2**

*Проектирование Веб-приложений. Прототипирование интерфейса.*

**УСЛОВИЕ:**

Разработать вайерфреймы или прототипы графического интерфейса портала согласно технического задания (ТЗ) и темы индивидуального задания. Требуется продумать и нарисовать следующие экраны / состояния приложения.

1. Разработать проект «Обработка заявок на авиабилеты». Каждая заявка содержит:

• Пунктотправки;

• Пункт назначения;

• Номер рейса;

• Дата вылета.

• ФИО пассажира;

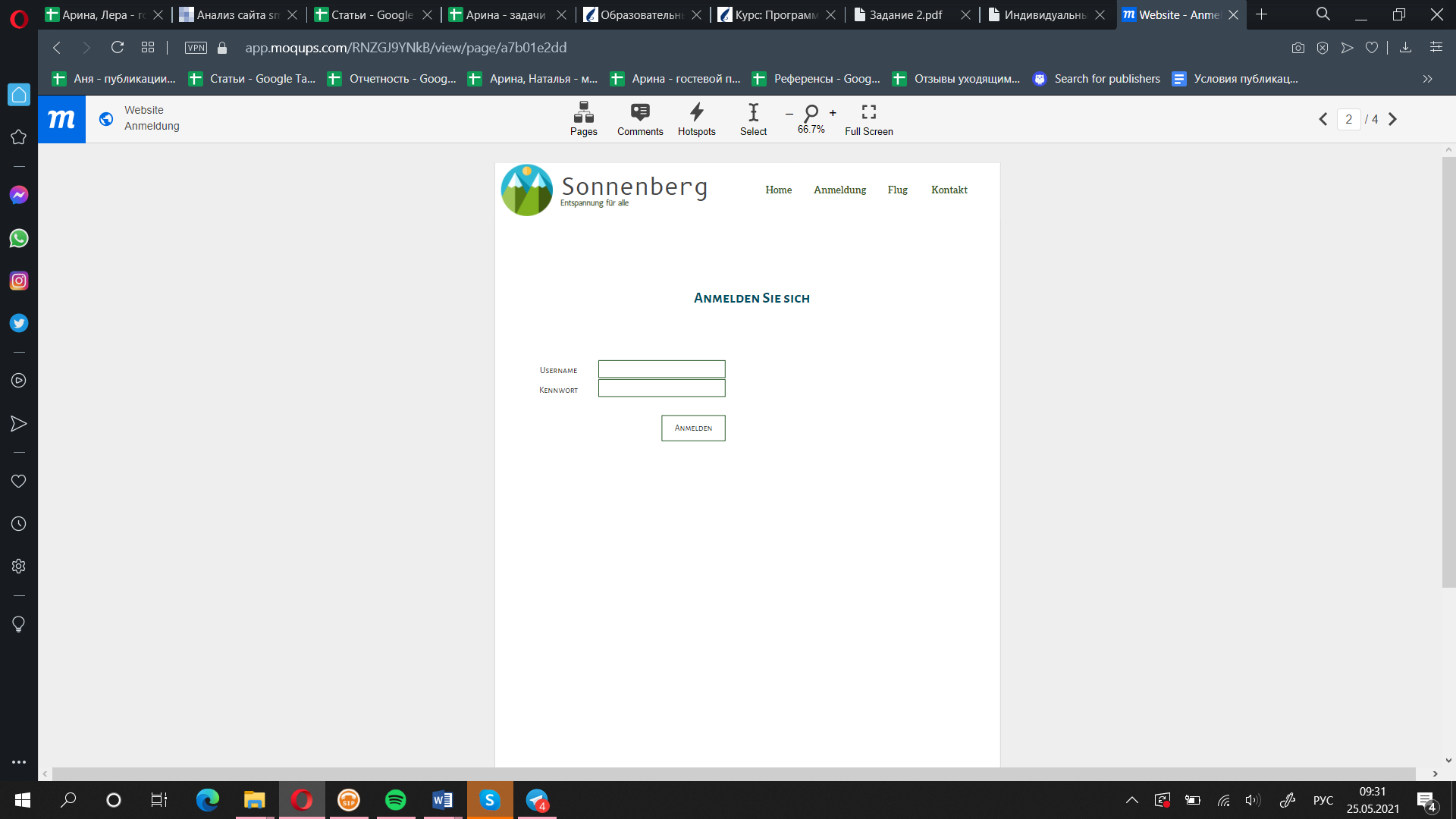
**РЕАЛИЗАЦИЯ:**

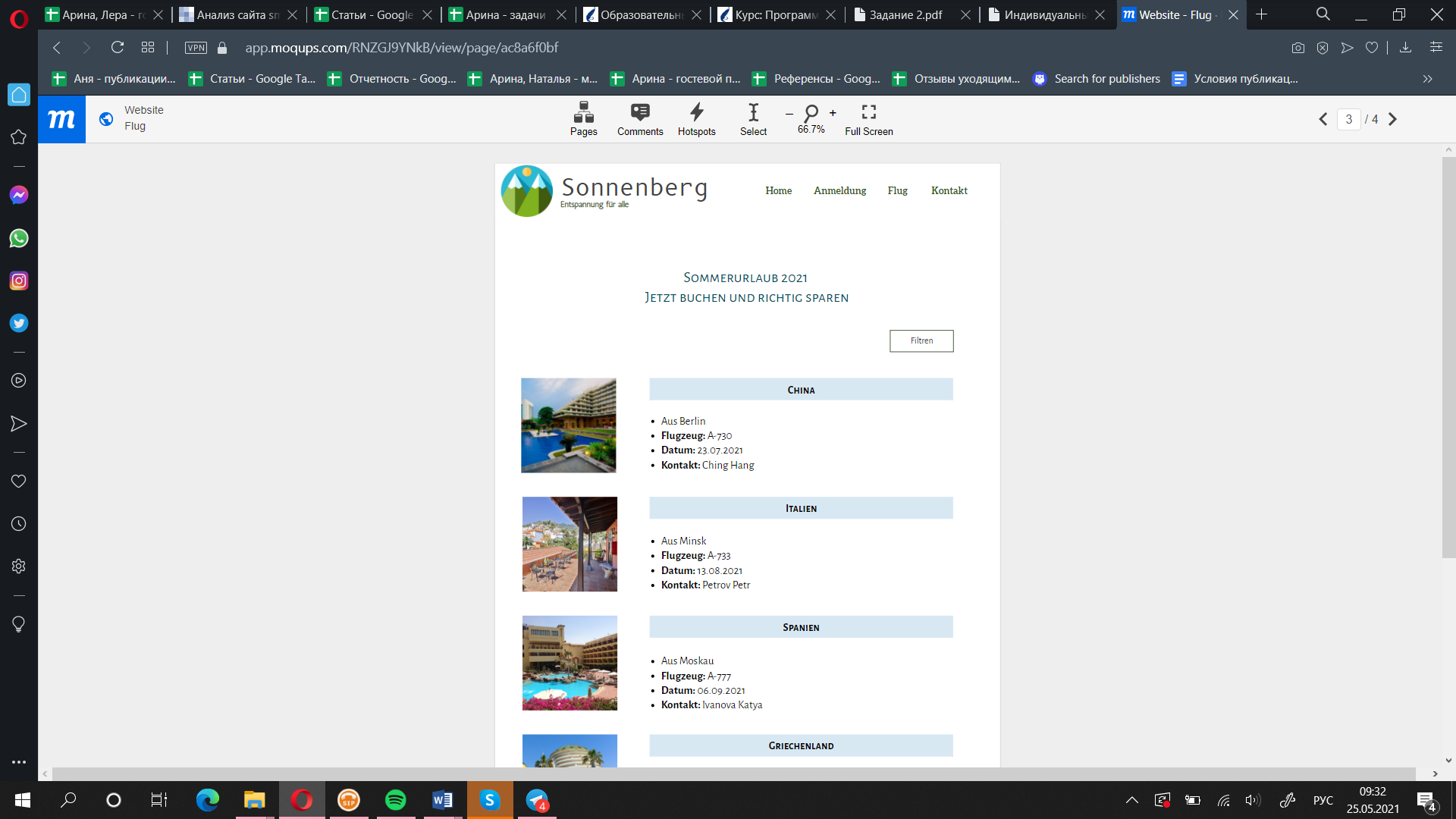
Использована среда moqups. Создан шаблон сайта из 4 страниц блочной верстки. Были использованы инструменты данного приложения для создания композиции. Картинки и текст были взяты с немецкого сайта, занимающегося перелетами. (lastminute.de).

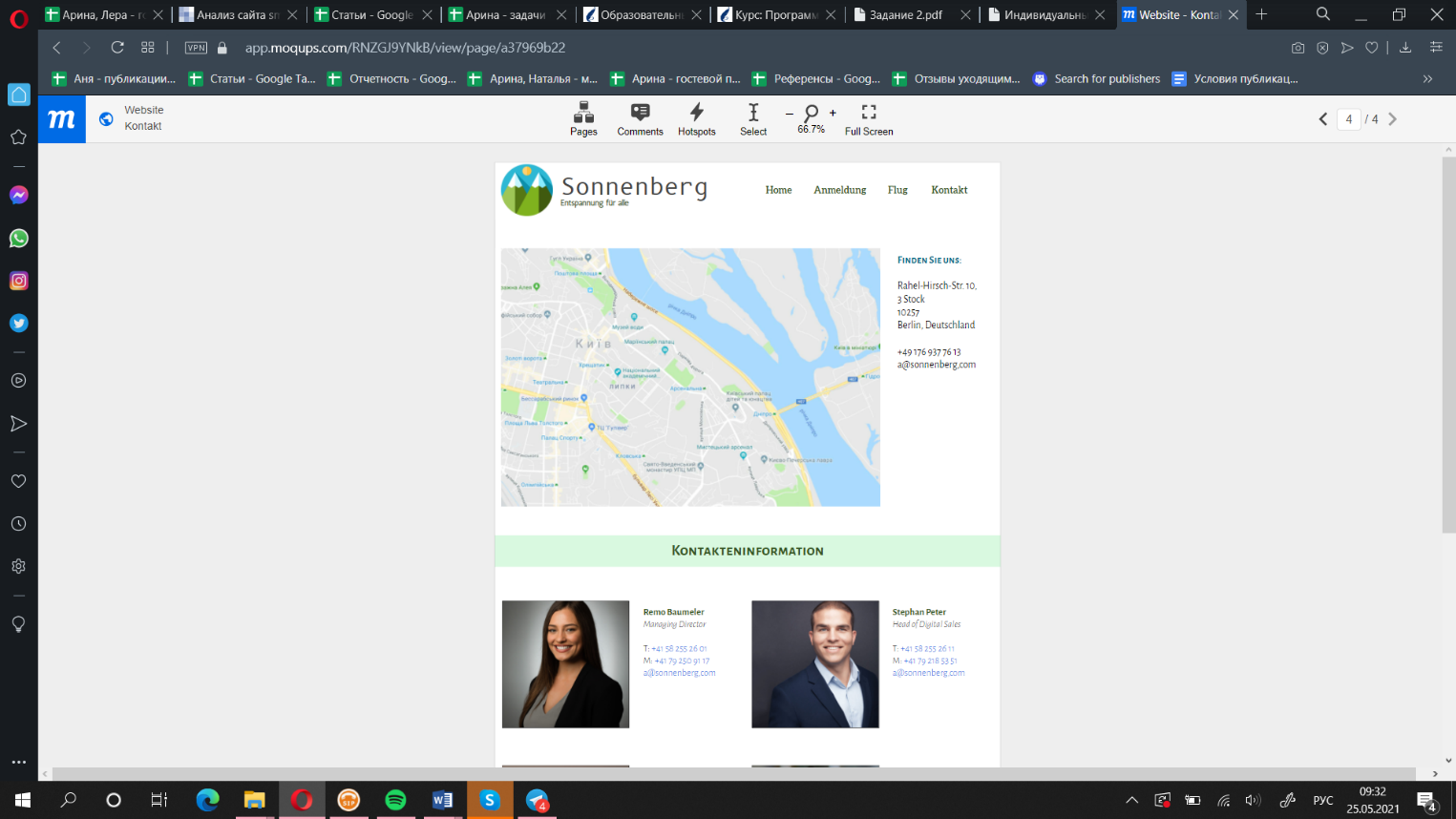
**РЕЗУЛЬТАТ ВЫПОЛНЕНИЯ:**

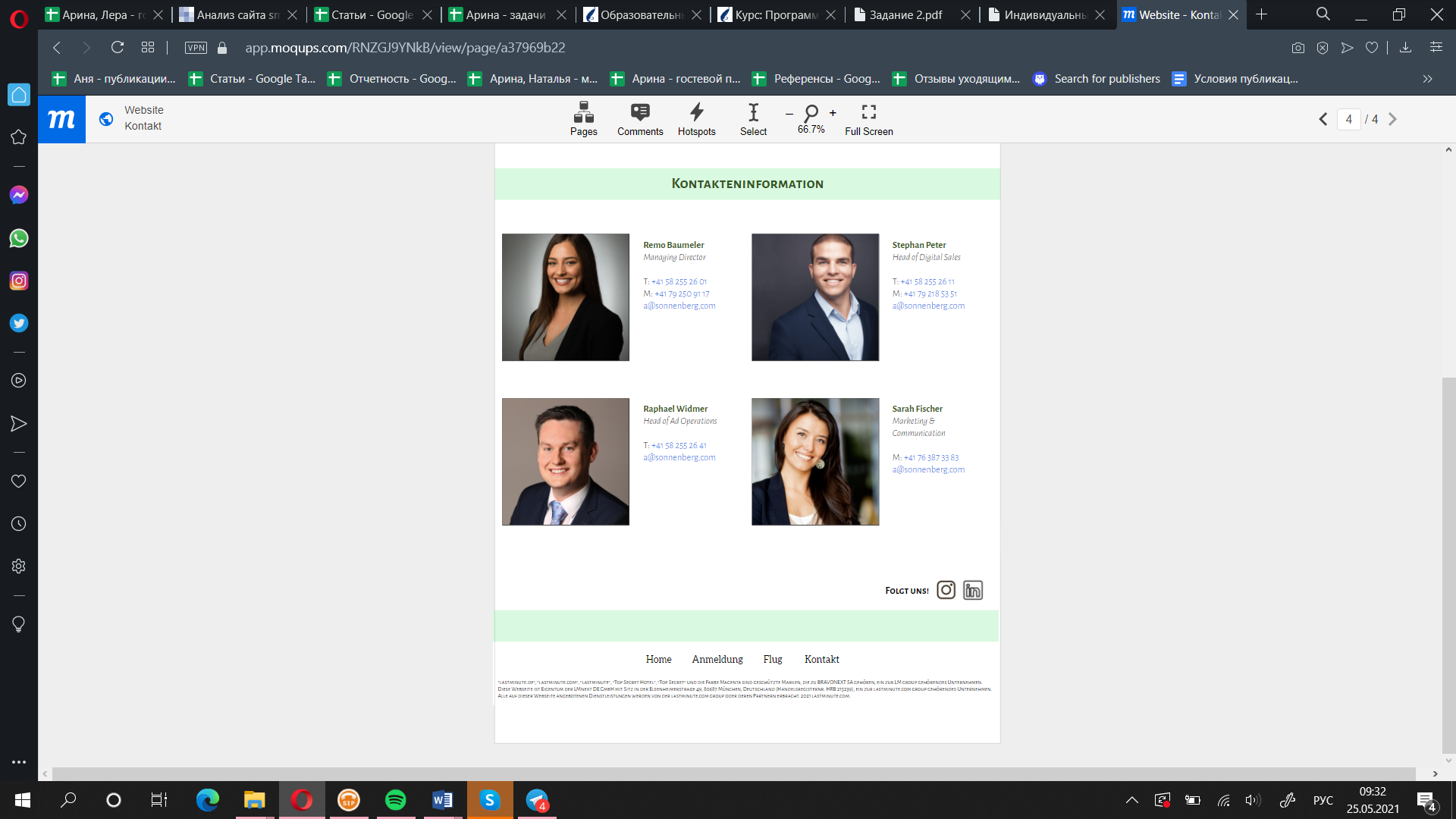
*(*[*https://app.moqups.com/RNZGJ9YNkB/view/page/ad64222d5*](https://app.moqups.com/RNZGJ9YNkB/view/page/ad64222d5)*)*

**

****

****

****

****

Идея и реализация дизайна принадлежит автору проекта.

**ЛАБОРАТОРНАЯ РАБОТА №3**

*Разработка интерфейса портала. Верстка.*

**УСЛОВИЕ:**

1. HTML + CSS Познакомиться с фундаментальными технологиями веб-разработки: HTML и CSS.
2. 2. Разработка интерфейса Используя HTML и CSS, разработать интерфейс главной страницы (экран просмотра ленты постов) web портала.

**РЕАЛИЗАЦИЯ:**

Верстка производилась в приложении «Блокнот».

<html>

<head>

<title>Sonnenberg</title>

<style>

body {

width: 900px;

height: 1500px;

margin: auto;

}

.header {

height: 92px;

}

p {

float: left;

}

.logo {

padding-left: 30px;

font-size: 13pt;

color:#355723;

font-family: Antic;

}

.links {

padding-top: 30px;

padding-left: 115px;

}

.links a {

padding-left: 45px;

color:#355723;

font-family: Aleo;

font-size: 13pt;

font-weight:bold;

text-decoration: none;

}

.line2

{

height:50px;

background-color: d9f9e0;

margin-top: 170px;

}

.links\_2 a

{

color: #000000;

text-decoration: none;

padding-left:45px;

font-weight:bold;

}

.links\_2 {

padding-left: 220px;

padding-top: 10px;

}

.rechts {

font-size:8pt;

color: #444444;

}

.stroke

{

margin-left:250px;

text-align:center;

font-family: Alegreya Sans SC;

font-size: 20pt;

color: #004e63;

}

.information

{

margin-top: 100px;

}

.button

{

margin-top: 250px;

margin-left:750px;

width:100px;

height:34px;

border: 2px solid #4c7a34;

text-align:center;

}

.filtren

{

margin-left: 30px;

margin-top:9px;

font-size:10pt;

font-family: Alegreya Sans;

}

.text {

width: 500px;

height: 35px;

background-color:#d8e8f3;

text-align:center;

font-family: Alegreya Sans SC;

font-size:14pt;

padding-top:7px;

}

table

{

margin-top: 50px;

}

td

{

padding-left: 50px;

}

li

{

font-family: Alegreya Sans;

font-size:14pt;

}

.folge

{

float: left;

height:50px;

padding-left:707px;

margin-top:105px;

}

.img

{

padding-bottom:7px;

}

</style>

</head>

<body>

<div class="header"><p class="headerr"><img src="10.png" width="82" height="82"><p class="logo"><font size="36px" color="#444444" face="B612 Mono">Sonnenberg</font><br />Entspannung für alle</p><p class="links"><a href="home.html">Home</a><a href="#">Anmeldung</a><a href="#">Flug</a><a href="#">Konkakt</a></p></p></div>

<div class="information"><p class="stroke">Sommerurlaub 2021<br />Jetzt buchen und richtig sparen!</p></div>

<div class="button"><p class="filtren">Filtren</p></div>

<div><table><tr><td><img src="6.jpg" width="170px" height="170px"></td><td><p class="text">China</p><br /><ul><li>Aus Berlin</li><li>Flugzeug: A-730</li><li>Datum: 23.07.2021</li><li>Kontakt: Ching Hang﻿</li></td></tr></table></div>

<div><table><tr><td><img src="12.jpg" width="170px" height="170px"></td><td><p class="text">Italien</p><br /><ul><li>Aus Minsk</li><li>Flugzeug: A-733</li><li>Datum: 13.08.2021</li><li>Kontakt: Petrov Petr﻿</li></td></tr></table></div>

<div><table><tr><td><img src="3.jpg" width="170px" height="170px"></td><td><p class="text">Spanien</p><br /><ul><li>Aus Moskau</li><li>Flugzeug: A-777</li><li>Datum: 06.09.2021</li><li>Kontakt: Ivanova Katya﻿</li></td></tr></table></div>

<div><table><tr><td><img src="18.jpg" width="170px" height="170px"></td><td><p class="text">Griechenland</p><br /><ul><li>Aus Riga</li><li>Flugzeug: A-321</li><li>Datum: 13.08.2021</li><li>Kontakt: Zaicev Vasiliy﻿</li></td></tr></table></div>

<div class="folge"><p>Folgt uns!</p><img src="7.jpg" width="50px" height="50px"><img src="13.png" width="35px" height="35px" class="img"></div>

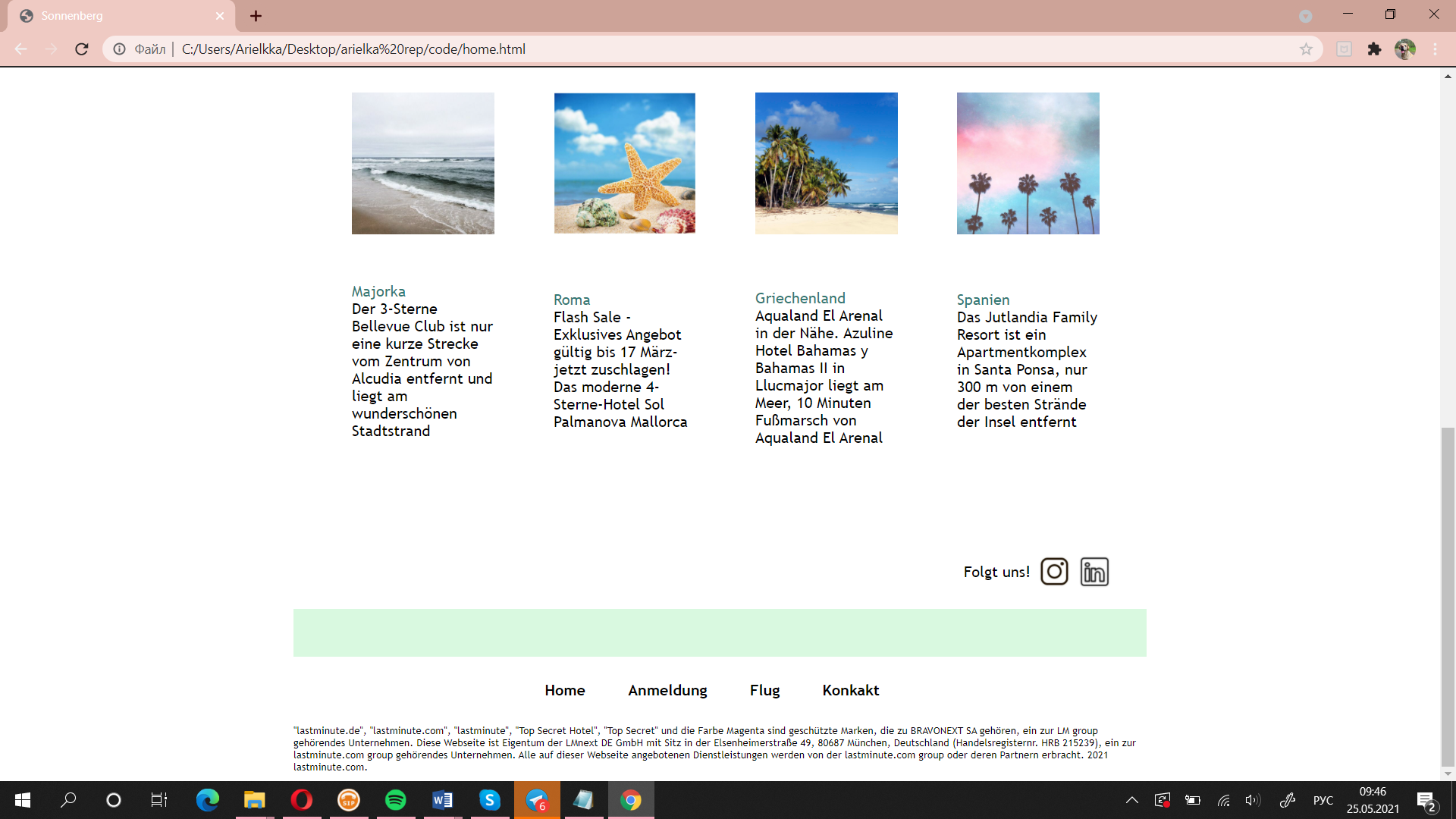
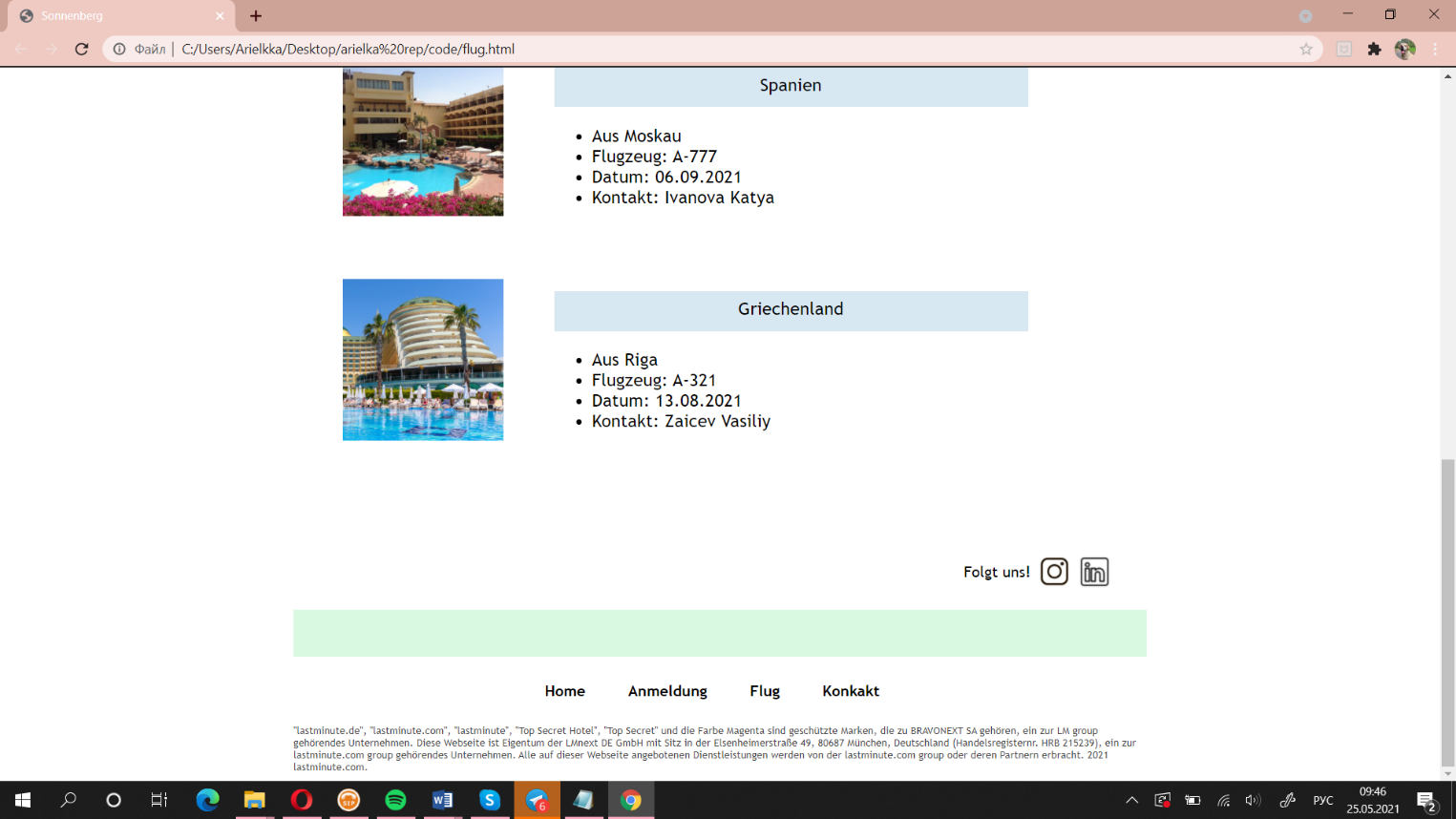
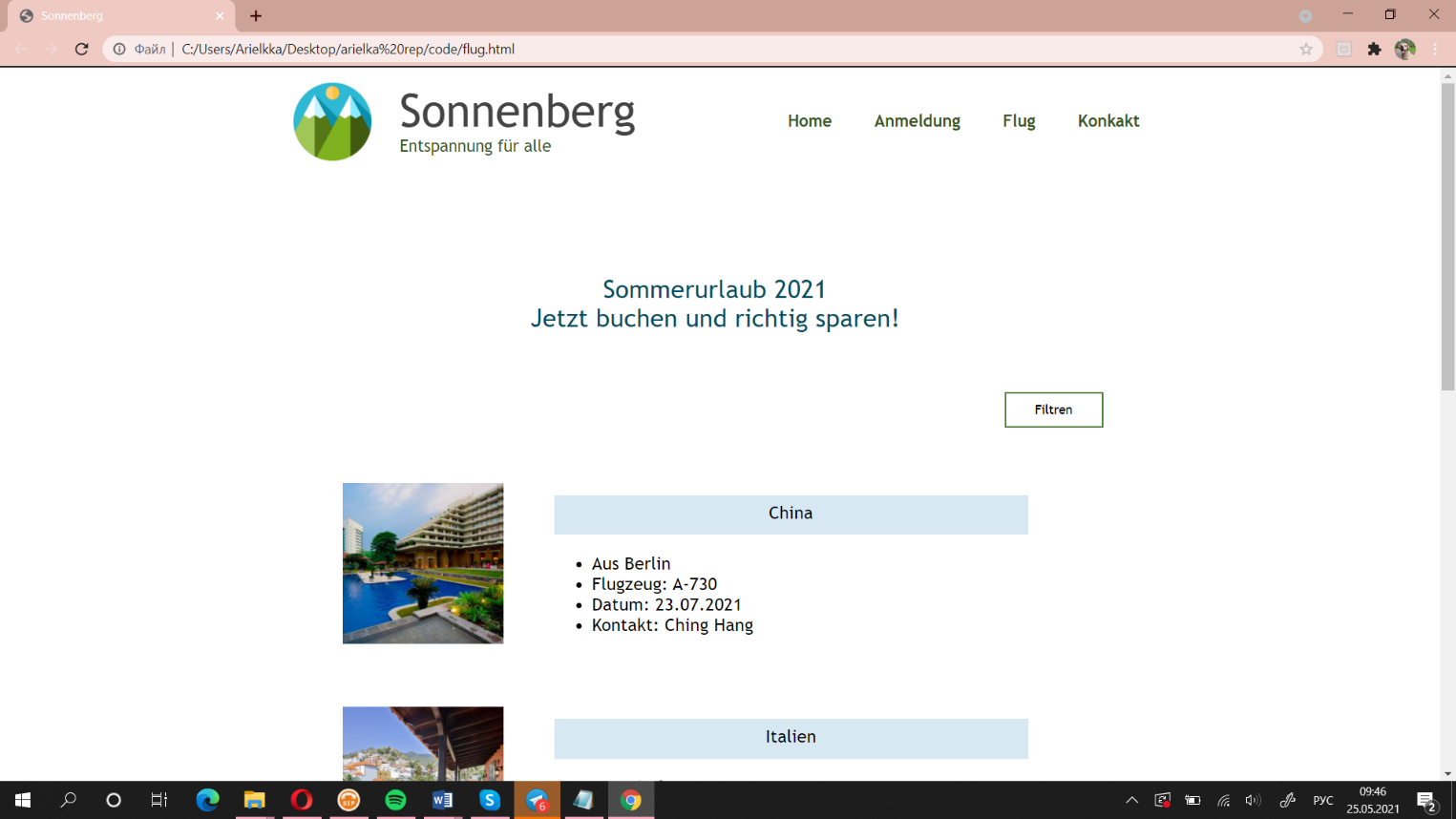
<div class="line2"></div>

<div class="footer"><p class="links\_2"><a href="#">Home</a><a href="#">Anmeldung</a><a href="#">Flug</a><a href="#">Konkakt</a></p></p><br /><p class="rechts">"lastminute.de", "lastminute.com", "lastminute", "Top Secret Hotel", "Top Secret" und die Farbe Magenta sind geschützte Marken, die zu BRAVONEXT SA gehören, ein zur LM group gehörendes Unternehmen. Diese Webseite ist Eigentum der LMnext DE GmbH mit Sitz in der Elsenheimerstraße 49, 80687 München, Deutschland (Handelsregisternr. HRB 215239), ein zur lastminute.com group gehörendes Unternehmen. Alle auf dieser Webseite angebotenen Dienstleistungen werden von der lastminute.com group oder deren Partnern erbracht. 2021 lastminute.com.</p></div>

</body>

</html>

**РЕЗУЛЬТАТ ВЫПОЛНЕНИЯ:**

****

**ЛАБОРАТОРНАЯ РАБОТА №4**

*Основы JavaScript. Синтаксис языка JavaScript. Функции, массивы, строки, сортировки. ES6 классы. MVC. Реализация функционала.*

**УСЛОВИЕ:**

1. Перенести данные из HTML в JavaScript. В JavaScript файле создать массив для хранения списка постов. Заполнить массив объектами Post

2. Перенести данные из HTML в JavaScript в класс, который должен содержать следующие методы для работы с массивом Posts

3. Правильно организовать приватные поля и методы

4. Дополнить класс недостающими методами.

5. Проверить работу всех методов. Требуется проверить работу всех методов. Выполнить их с различными параметрами: валидными и не валидными.

**РЕАЛИЗАЦИЯ:**

var posts = [  
 {  
 id: '1',  
 destination: 'China',  
 createdAt: new Date(2014, -1, 1, 2, 3, 4, 567),  
 author: 'ChingHang',  
 tag: "summer",  
 flug: 'A-730'  
 },  
 {  
 id: '2',  
 destination: 'Italien',  
 createdAt: new Date(2014, -1, 1, 2, 3, 4, 567),  
 author: 'PetrovPetr',  
 tag: "summer",  
 flug: 'A-733'  
 },  
 {  
 id: '3',  
 destination: 'Spanien',  
 createdAt: new Date(2017, 0, 1, 2, 3, 4, 567),  
 author: 'IvanovaKatya',  
 tag: "spring",  
 flug: 'A-777'  
 },  
 {  
 id: '4',  
 destination: 'Griechenland',  
 createdAt: new Date(2013, 0, 1, 2, 3, 4, 567),  
 author: 'ZaicevVasiliy',  
 tag: "spring",  
 flug: 'A-321'  
 },  
];

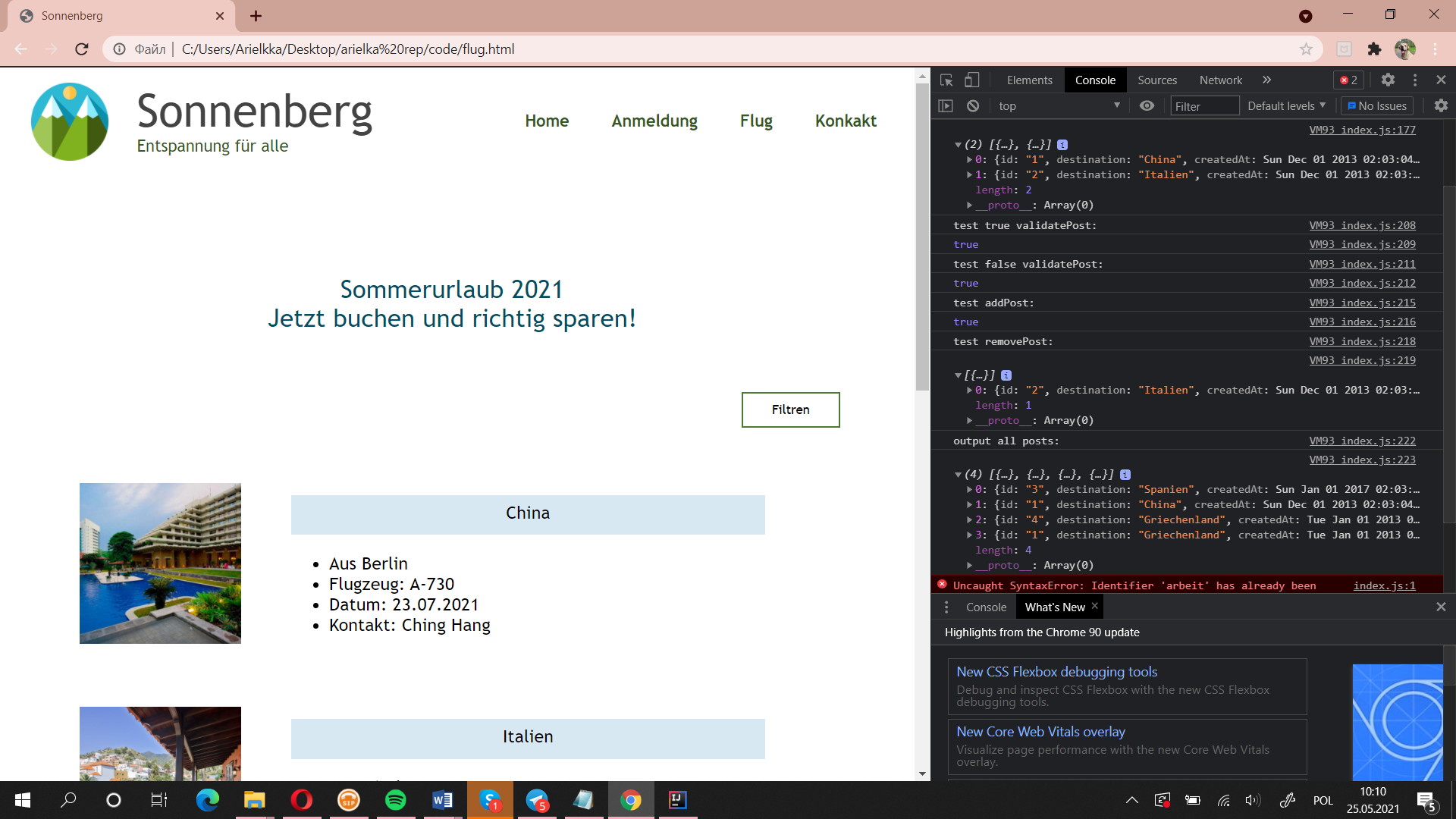
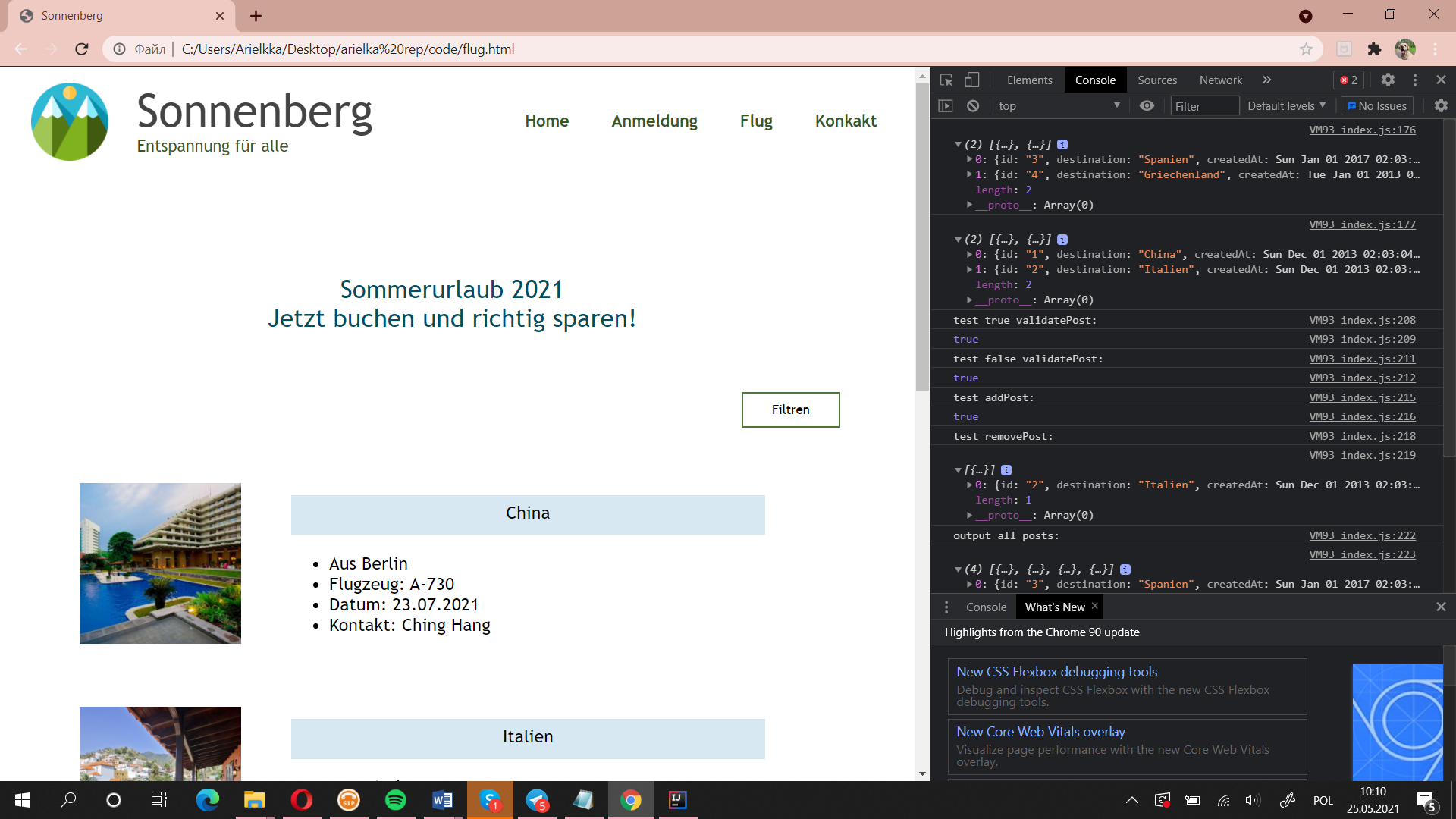
class arbeit {  
 getPosts(filterConfig = "Date", skip = 0, top = posts.length) {  
 if (filterConfig == "Date") {  
 posts.sort(function (a, b) {  
 return new Date(b.createdAt) - new Date(a.createdAt);  
 });  
 return posts.slice(skip, top + skip);  
 }  
 else if (filterConfig == "summer") {  
 var temp = []  
 posts.forEach((product, ind) => {  
 if (product.tag == "summer"){  
 temp.push(product)  
 }  
 });  
 return temp  
 }  
 else if (filterConfig == "spring") {  
 var temp = []  
 posts.forEach((product, ind) => {  
 if (product.tag == "spring"){  
 temp.push(product)  
 }  
 });  
 return temp  
 }  
 else {  
 throw "Error. there is no such filter";  
 }  
 }  
 getPost(id) {  
 for (var i = 0; i < posts.length; i++) {  
 if (posts[i].id === id) {  
 return posts[i];  
 }  
 }  
 throw "No object with " + id + " id";  
 }  
 validatePost(Object){  
 return Object.id != null && Object.destination != null && Object.destination.length<400  
 && Object.author != null && Object.createdAt != null && typeof Object.id === "string" && typeof Object.destination === "string" && typeof Object.flug === "string"  
 && typeof Object.author === "string"  
  
 }  
 addAll(posts\_double){  
 let result=[], k=0;  
 for(let i = 0; i<posts\_double.length; i++){  
 if(this.validatePost(posts\_double[i])){  
 this.addPost(posts\_double[i])  
 }else {  
 result[k] = posts\_double[i];  
 k++;  
 }  
 }  
 return result;  
 }  
 addPost(Object){  
 if(this.validatePost(Object)){  
 posts.splice(posts.length,0,Object);  
 return true;  
 }  
 else {  
 return false;  
 }  
 }  
 clear(){  
 posts.splice(0,posts.length);  
 }  
 editPost(id,post){  
 if(this.validatePost(post)){  
 this.getPost(id).destination=post.destination;  
 this.getPost(id).author=post.author;  
 return true;  
 }else {  
 return false;  
 }  
 }  
 removePost(id){  
 for (var i = 0; i < posts.length; i++) {  
 if (posts[i].id === id) {  
 return posts.splice(i,1);  
 }  
 }  
 throw "No object with " + id + " id";  
 }  
 constructor(posts) {  
 this.\_posts=posts;  
 }  
}

**Обработка данных:**

let a = new arbeit(posts);  
  
let ob={  
 id: '5',  
 destination: 'England',  
 createdAt: new Date(2018, 0, 1, 2, 3, 4, 567),  
 author: 'ZaicevAnatoliy',  
 hashTags:['Brest'],  
 flug: 'A-737'  
}  
  
posts\_true = {  
 id: '1',  
 destination: 'Griechenland',  
 createdAt: new Date(2013, 0, 1, 2, 3, 4, 567),  
 author: 'SergeevaAnna',  
 flug: 'A-322'  
 }  
posts\_false = {  
 id: '12',  
 destination: 'Frankreich',  
 createdAt: new Date(2019, 0, 1, 2, 3, 4, 567),  
 author: 'KotovaMariya',  
 flug: 'A-321'  
 }  
  
  
console.log(a.getPosts('spring'))  
console.log(a.getPosts('summer'))  
  
console.log(a.printProduct(a.getPosts('spring', 0, 1)))  
  
console.log("testing getPost: ")  
console.log(a.getPost(3))

console.log("test true validatePost: ")  
console.log(a.validatePost(posts\_true))  
  
console.log("test false validatePost: ")  
console.log(a.validatePost(posts\_false))  
  
  
console.log("test addPost: ")  
console.log(a.addPost(posts\_true))  
  
console.log("test removePost: ")  
console.log(a.removePost("2"))  
  
  
console.log("output all posts: ")  
console.log(a.getPosts("Date"))

**РЕЗУЛЬТАТ ВЫПОЛНЕНИЯ:**

****

**ЛАБОРАТОРНАЯ РАБОТА №5**

*Работа с DOM*

**УСЛОВИЕ:**

1. Отобразить данные с помощью JavaScript и DOM API
2. Реализовать класс View
3. Реализовать функции для демонстрации работы бизнес-логики приложения

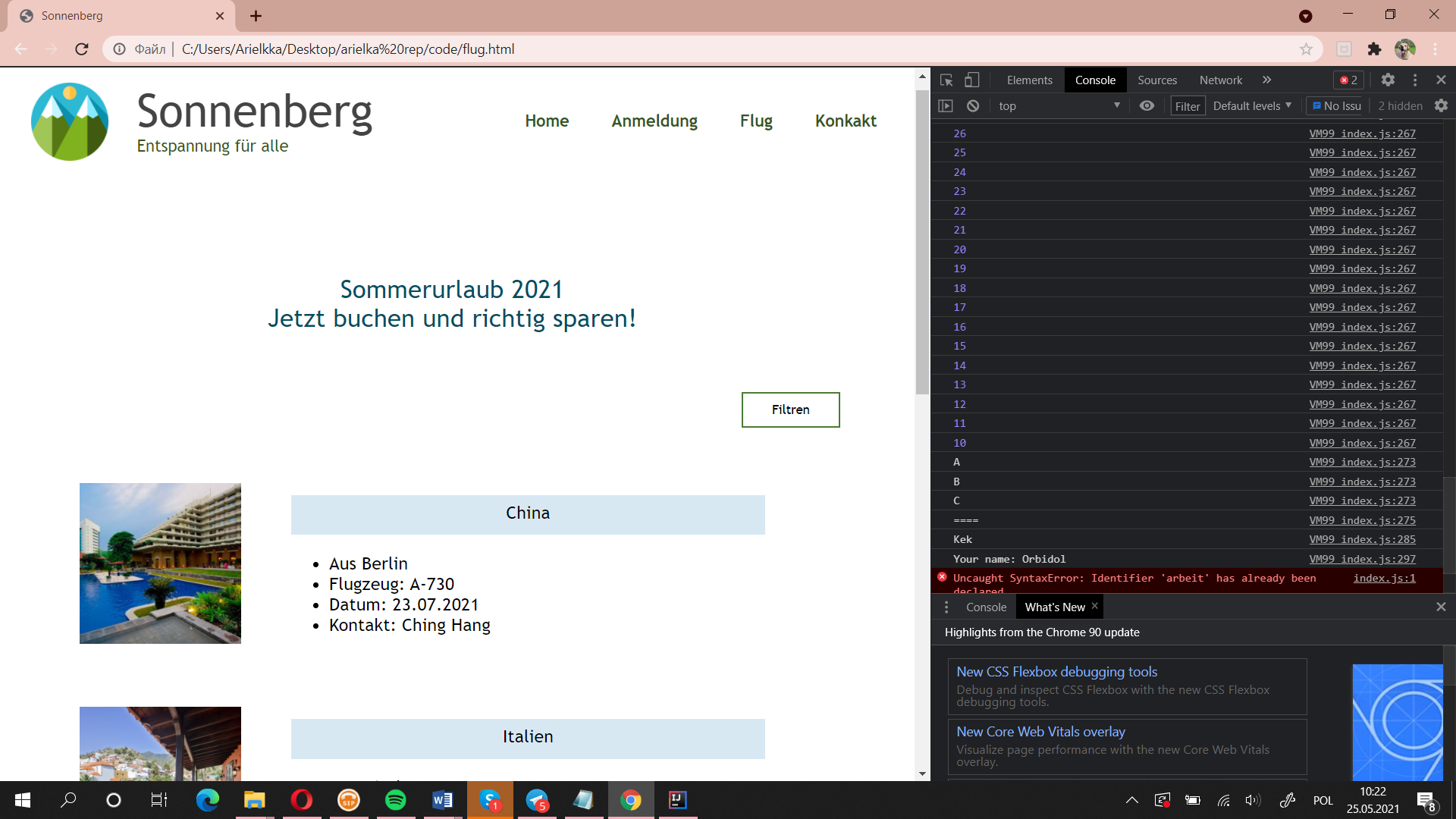
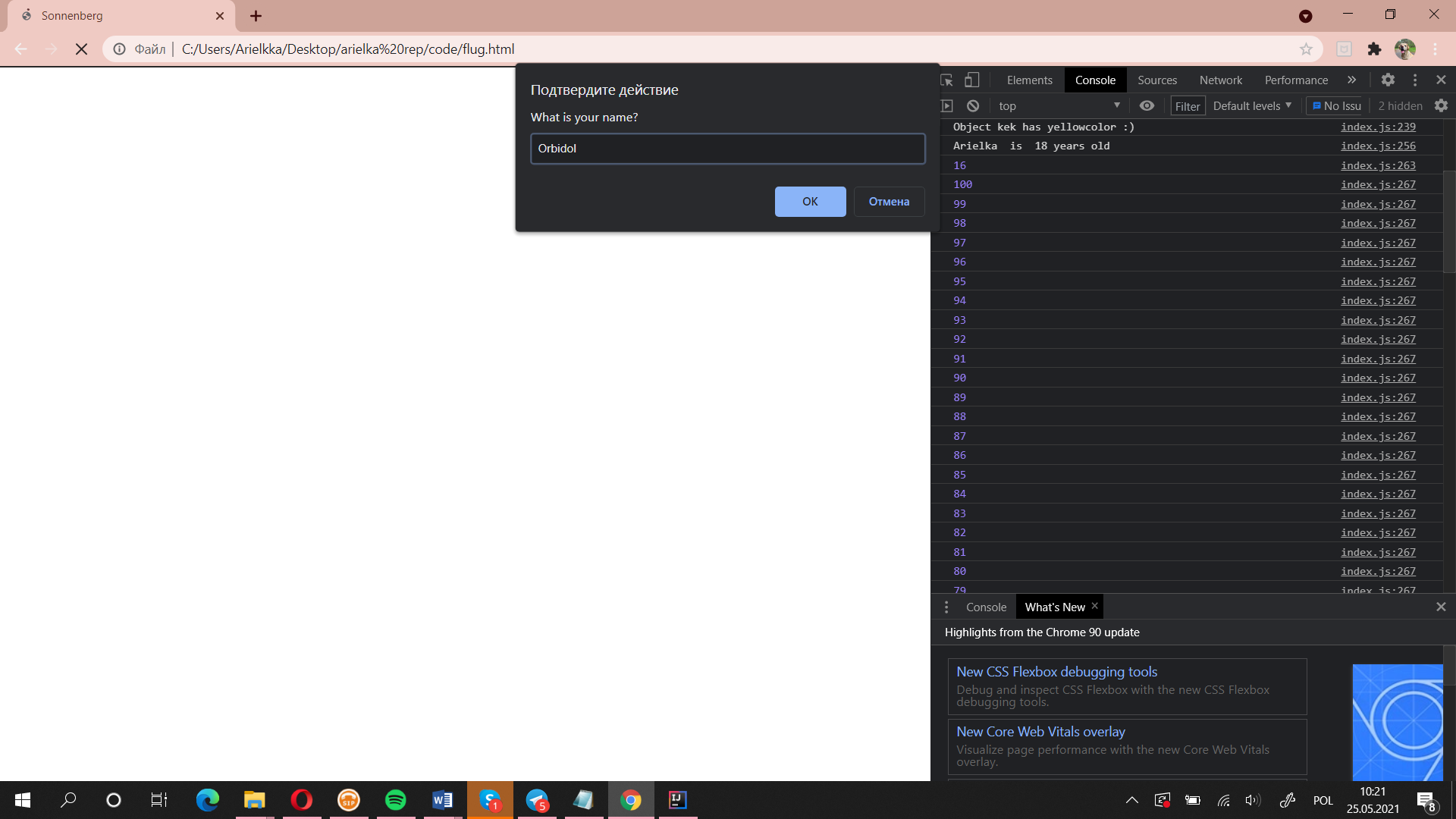
**РЕАЛИЗАЦИЯ:**

class arbeit {  
 printProduct(data) {  
 div\_element.innerHTML = "";  
 data.forEach((product, ind) => {  
 div\_element.innerHTML +=`  
 <table>  
 <tr>  
 <td>  
 <img src="6.jpg" width="170px" height="170px">  
 </td>  
 <td>  
 <p class="text">${product.destination}</p>  
 <br />  
 <ul>  
 <li>Aus Berlin</li>  
 <li>${product.flug}</li>  
 <li>${product.createAt}</li>  
 <li>${product.author}</li>  
 </td>  
 </tr>  
 </table>  
 `  
 });  
 }

**Остальной скрипт класса не отличается. Обработка:**

console.log(a.printProduct(a.getPosts('spring', 0, 1)))  
  
  
function printProduct() {  
 posts.forEach((product, ind) => {  
 document.write(`  
<table>  
<tr>  
<td>  
<img src="6.jpg" width="170px" height="170px">  
</td>  
<td>  
<p class="text">${product.destination}</p>  
<br />  
<ul>  
<li>Aus Berlin</li>  
<li>${product.flug}</li>  
<li>${product.createAt}</li>  
<li>${product.author}</li>  
</td>  
</tr>  
</table>  
 `)  
 });  
 }  
  
  
function changeColor(newColor) {  
 var element = document.getElementById("text");  
 element.style.color = newColor;  
}  
  
  
  
 var kek = new Object();  
 kek.color = "yellow";  
 kek.shape = "round";  
  
 kek.describe = function() {  
 return "Object kek has " + this.color + "color :)";  
 }  
  
 console.log(kek.describe());  
  
  
  
var user = {  
 firstName: "Arielka",  
 age: 18,  
 children: ["10", "20"],  
 adress: {  
 street: "Plekhanova Str. 61",  
 city: "Serebruklin"  
 },  
 Name: function() {  
 return this.firstName + " is " + this.age + " years old";  
 }  
}  
  
console.log(user.Name());  
  
  
function sum (a,b) {  
 return a + b  
}  
  
console.log(sum(12,4));  
  
  
for(var i = 100; i >=10; i--) {  
 console.log(i);  
 }  
  
  
var mas = ["A", "B", "C"];  
for(var i = 0; i < mas.length; i++){  
 console.log(mas[i]);  
 }  
 console.log("====");  
  
  
var first = 18;  
var second = 25;  
  
var structure = [1,2,3,4];  
var structure\_1 = new Array (1,2,3,4,5)  
structure.push("Kek")  
  
console.log(structure[4]);  
  
  
document.write("result of concat: " + first + second);  
document.write("result of summing: ", first + second);  
  
  
  
var num\_1 = 22;  
var num\_2 = 11;  
  
var name = prompt("What is your name?");  
console.log("Your name: " + name);

**РЕЗУЛЬТАТ ВЫПОЛНЕНИЯ:**

****

**ЛАБОРАТОРНАЯ РАБОТА №7**

*Java Servlets*

**УСЛОВИЕ:**

1. Настроить рабочую среду для работы java & web

2. Разобраться с работой java servlets

3. Создать простое web-приложение с использованием сервлетов

**РЕАЛИЗАЦИЯ:**

*ChechServlet.java*

import javax.servlet.ServletException;  
import javax.servlet.annotation.WebServlet;  
import javax.servlet.http.HttpServlet;  
import javax.servlet.http.HttpServletRequest;  
import javax.servlet.http.HttpServletResponse;  
import java.io.IOException;  
  
@WebServlet(name = "checkServlet", value = "/check")  
public class CheckServlet extends HttpServlet {  
 @Override  
 protected void doPost(HttpServletRequest request, HttpServletResponse response) throws ServletException, IOException, IOException {  
 response.setContentType("json");  
 response.getWriter().println("{\"success\" : \"true\"}");  
 }  
  
 @Override  
 protected void doGet(HttpServletRequest request, HttpServletResponse response) throws ServletException, IOException {  
 response.sendRedirect("/demo\_war");  
 }  
}

*ForwardServlet.java*

import javax.servlet.ServletException;  
import javax.servlet.annotation.WebServlet;  
import javax.servlet.http.HttpServlet;  
import javax.servlet.http.HttpServletRequest;  
import javax.servlet.http.HttpServletResponse;  
import java.io.IOException;  
  
@WebServlet(name = "forwardServlet", value = "/test1")  
public class ForwardServlet extends HttpServlet {  
 @Override  
 protected void doGet(HttpServletRequest request, HttpServletResponse response) throws ServletException, IOException {  
 request.getRequestDispatcher("/status").forward(request, response);  
 }  
}

*GetServlet.java*

import javax.servlet.ServletException;  
import javax.servlet.annotation.WebServlet;  
import javax.servlet.http.HttpServlet;  
import javax.servlet.http.HttpServletRequest;  
import javax.servlet.http.HttpServletResponse;  
import java.io.IOException;  
import java.io.PrintWriter;  
  
@WebServlet(name = "getServlet", value = "/get")  
public class GetServlet extends HttpServlet {  
 @Override  
 protected void doGet(HttpServletRequest request, HttpServletResponse response) throws ServletException, IOException {  
 String name = request.getParameter("name");  
  
 PrintWriter out = response.getWriter();  
 out.println("<html><body>");  
 out.println("<h3> Name is " + name + "</h3>");  
 out.println("</body></html>");  
 }  
}

*PageServlet.java*

import javax.servlet.ServletException;  
import javax.servlet.annotation.WebServlet;  
import javax.servlet.http.HttpServlet;  
import javax.servlet.http.HttpServletRequest;  
import javax.servlet.http.HttpServletResponse;  
import java.io.\*;  
  
@WebServlet(name = "pageServlet", value = "/page")  
public class PageServlet extends HttpServlet {  
 @Override  
 public void doGet(HttpServletRequest request, HttpServletResponse response) throws ServletException, IOException {  
 request.getRequestDispatcher("/WEB-INF/page.html").forward(request, response);  
 }  
}

*RedirectServlet.java*

import javax.servlet.\*;  
import javax.servlet.http.\*;  
import javax.servlet.annotation.\*;  
import java.io.IOException;  
  
@WebServlet(name = "redirectServlet", value = "/test2")  
public class RedirectServlet extends HttpServlet {  
  
 @Override  
 protected void doGet(HttpServletRequest request, HttpServletResponse response) throws ServletException, IOException {  
 request.getRequestDispatcher("/page").forward(request, response);  
 }  
}

*StatusServlet.java*

import javax.servlet.annotation.WebServlet;  
import javax.servlet.http.\*;  
import java.io.\*;  
  
@WebServlet(name = "statusServlet", value = "/status")  
public class StatusServlet extends HttpServlet {  
 private static final String announce = "It works";  
  
 public void doGet(HttpServletRequest request, HttpServletResponse response) throws IOException {  
 response.setContentType("text/html");  
  
 PrintWriter out = response.getWriter();  
 out.println("<html><body>");  
 out.println("<h3 style = 'color: green'>" + announce + "</h3>");  
 out.println("</body></html>");  
 }  
}

*TimeMeasurementFilter.java*

import javax.servlet.\*;  
import javax.servlet.annotation.\*;  
import javax.servlet.http.HttpServletRequest;  
import java.io.IOException;  
  
@WebFilter(filterName = "timeMeasurementFilter")  
public class TimeMeasurementFilter implements Filter {  
 public void init(FilterConfig config) throws ServletException {  
 }  
  
 public void destroy() {  
 }  
  
 @Override  
 public void doFilter(ServletRequest request, ServletResponse response, FilterChain chain) throws ServletException, IOException {  
 long start = System.currentTimeMillis();  
 chain.doFilter(request, response);  
 long end = System.currentTimeMillis();  
  
 HttpServletRequest httpRequest = (HttpServletRequest) request;  
 String path = httpRequest.getRequestURI();  
 String method = httpRequest.getMethod();  
  
 System.out.println(String.format(("%s '%s' - ended (%d ms)"), method, path, end-start));  
 }  
}

**ИСПОЛЬЗОВАННАЯ ЛИТЕРАТУРА**

1. Основы компьютерных сетей. Передача данных. Cетевой протокол HTTP
2. Введение. DOM в примерах
3. DOM: работа с HTML-страницей
4. Программирование на JavaScript
5. Основы JavaScript
6. Основы JavaScript для начинающих разработчиков
7. Основы HTML и CSS
8. Мокап, вайрфрейм, прототип и скетч: ху из ху
9. Принцип золотого сечения в дизайне сайтов
10. Введение в события
11. Формы, элементы управления