

МIНIСТЕРСТВО ОСВIТИ І НАУКИ УКРАЇНИ

НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ УКРАЇНИ

“КИЇВСЬКИЙ ПОЛІТЕХНІЧНИЙ ІНСТИТУТ

ІМЕНІ ІГОРЯ СІКОРСЬКОГО”

Факультет прикладної математики

Кафедра програмного забезпечення комп’ютерних систем

**Лабораторна робота №** **9**

з дисципліни “ Основи програмування ”

тема “Браузерний JavaScript та AJAX”

|  |  |  |
| --- | --- | --- |
| Виконав  студент II курсу  групи КП-83  Палій Дмитро Володимирович  (*прізвище, ім’я, по батькові*)  варіант № 19 |  | Перевірив  “\_\_\_\_” “\_\_\_\_\_\_\_\_\_\_\_\_” 20\_\_\_ р.  викладач  Гадиняк Руслан Анатолійович  (*прізвище, ім’я, по батькові*) |

Київ 2019

**Мета роботи**

Навчитись створювати динамічні веб-сторінки за допомогою браузерних скриптових мов.

Навчитись працювати із технологією AJAX для отримання даних від REST API веб-сервера.

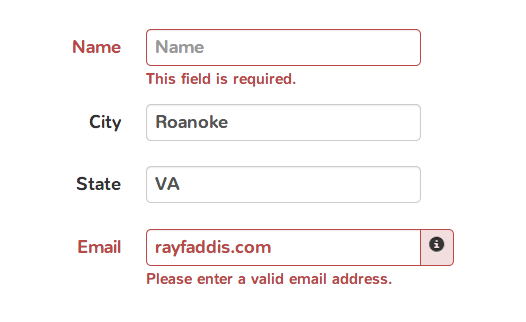
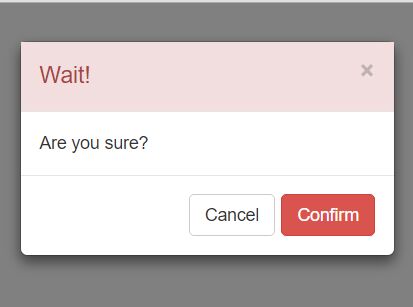
**Постановка завдання**

### Підготовка

Потрібно:

1. Ініціалізація проекту завдання
   1. Створити директорію labs/lab9 в репозиторії A і перейти до неї.  
      Всередині розмістити код web-сервера із попередніх завдань.
2. Створити у директорії public директорію javascripts

### Попереднє завдання

1. HTML та стилі
   1. Додати на сайт сторінку для відповідей зі статусом 404 Not found.
   2. Додати уніфіковані стилі до всіх сторінок. Стилізувати всі HTML елементи.  
      Рекомендується використати Bootstrap або аналогічний фреймворк.
   3. Додати спеціальні HTML-атрибути для клієнтської валідації всіх полів вводу на формах сайту.  
      Приклад форми із валідацією:  
      
2. Використання браузерного JS:
   1. Додати до дій видалення сутностей підтвердження за допомогою стилізованих модальних вікон.  
      Приклад модального вікна видалення елемента:  
        
      Використовувати стандартне вікно браузера (confirm()) не можна.  
      Використайте модальні вікна Bootstrap або опишіть власні елемени для модального вікна.

### Завдання

Реалізувати:

1. На сторінці перегляду і пошуку сутностей:
   1. **Пошук**: реалізувати AJAX запит до REST API сервера із динамічним оновленням вмісту поточної web-сторінки та відображення результатів пошуку.
   2. **Пагінація**: реалізувати AJAX запити із динамічним оновленням вмісту поточної web-сторінки та відображення сторінки даних.

Використати:

* чистий браузерний JavaScript та AJAX (або **FetchAPI**, але без third-party залежностей, таких як JQuery чи axios)
* браузерний шаблонізатор рядків (наприклад, **Mustache**) для створення нових DOM-елементів сторінки браузерним JavaScript. Шаблони зробити зовнішніми ресурсами і завантажувати з браузерного JavaScript.
* для авторизації *деяких\** запитів до REST API з попереднього завдання додати можливість перевірки **сесії**, або **JWT**.  
  (*\* якщо у завданні потрібно виконувати доступ до endpoint з авторизацією*)

**Тексти коду програм**

|  |
| --- |
| **client.js** |
| const jwt = localStorage.getItem('jwt');  if (jwt) {  Promise.all([  fetch('/api/v1/me', { headers: { Authorization: `Bearer ${jwt}` }}).then(res => {  if(res.status == 404) {  window.location.replace("/notfound");  return Promise.reject('Not Found');  }  else return res;  }).then(x => x.json()),  fetch('/templates/auth.mst').then(x => x.text())  ])  .then(([data, template]) => {  if (data.user) {  const renderedHtmlStr = Mustache.render(template, data);  $("#categories").removeClass('d-none');  $("#products").removeClass('d-none');  if (data.user.role === 'admin') $("#users").removeClass('d-none');  return renderedHtmlStr;  }  else return Promise.reject({ err: 'not logged in'});  })  .then(htmlStr => {  const el = document.getElementById('auth');  el.innerHTML = htmlStr;  const logoutForm = document.getElementById("logoutLink");  logoutForm.addEventListener('click', onLogout);  })  .catch(err => console.error(err));  }  else {  fetch('/templates/noauth.mst').then(x => x.text())  .then((template) => {  return Mustache.render(template);  })  .then(htmlStr => {  const el = document.getElementById('auth');  el.innerHTML = htmlStr;  const login = document.getElementById("loginBtn");  login.addEventListener('click', onLogin);  const register = document.getElementById("registerBtn");  register.addEventListener('click', onRegister);  })  .catch(err => console.error(err));  }  function onLogout(e) {  e.preventDefault();  localStorage.removeItem('jwt');  window.location.replace("/");  }  function onLogin(e) {  const loginForm = document.getElementById("login-form");  loginForm.addEventListener('submit', function(event) {  event.preventDefault();  const formData = new FormData(event.target);  const bodyData = new URLSearchParams(formData);  const error = document.getElementById('login-error');  if (loginForm.username.value == '' || loginForm.password.value == '') {  event.stopPropagation();  $("#login-error").removeClass('d-none');  error.innerHTML = 'Please enter username and password';  }  else {  fetch("/api/v1/auth/", {  method: 'POST',  headers: {  'Content-Type': 'application/x-www-form-urlencoded;charset=UTF-8',  'Authorization': `Bearer ${jwt}`  },  body: bodyData  })  .then(x => x.json())  .then(authResult => {  if (authResult.err) {  $("#login-error").removeClass('d-none');  error.innerHTML = authResult.err;  }  else {  $("#login-error").addClass('d-none');  const jwt = authResult.token;  localStorage.setItem("jwt", jwt);  $("#loginModal").modal('hide');  document.location.reload(true);  }  })  .catch(console.error);  }  });  }  function onRegister(e) {  const registerForm = document.getElementById('register-form');  registerForm.addEventListener('submit', function(event) {  event.preventDefault();  const formData = new FormData(event.target);  const bodyData = new URLSearchParams(formData);  const error = document.getElementById('register-error');  if (registerForm.username.value == '' || registerForm.password.value == '') {  event.stopPropagation();  $("#register-error").removeClass('d-none');  error.innerHTML = 'Please enter username and password';  }  else if (registerForm.password.value !== registerForm.confirm.value) {  event.stopPropagation();  $("#register-error").removeClass('d-none');  error.innerHTML = 'Passwords do not match';  }  else {  fetch("/api/v1/users", {  method: 'POST',  headers: {  'Content-Type': 'application/x-www-form-urlencoded;charset=UTF-8',  'Authorization': `Bearer ${jwt}`  },  body: bodyData  })  .then(x => x.json())  .then(regResult => {  if (regResult.err) {  $("#register-error").removeClass('d-none');  error.innerHTML = regResult.err;  return Promise.reject(regResult.err);  }  else {  $("#register-error").addClass('d-none');  $("#registerModal").modal('hide');  }  })  .catch(console.error);  }  });  } |

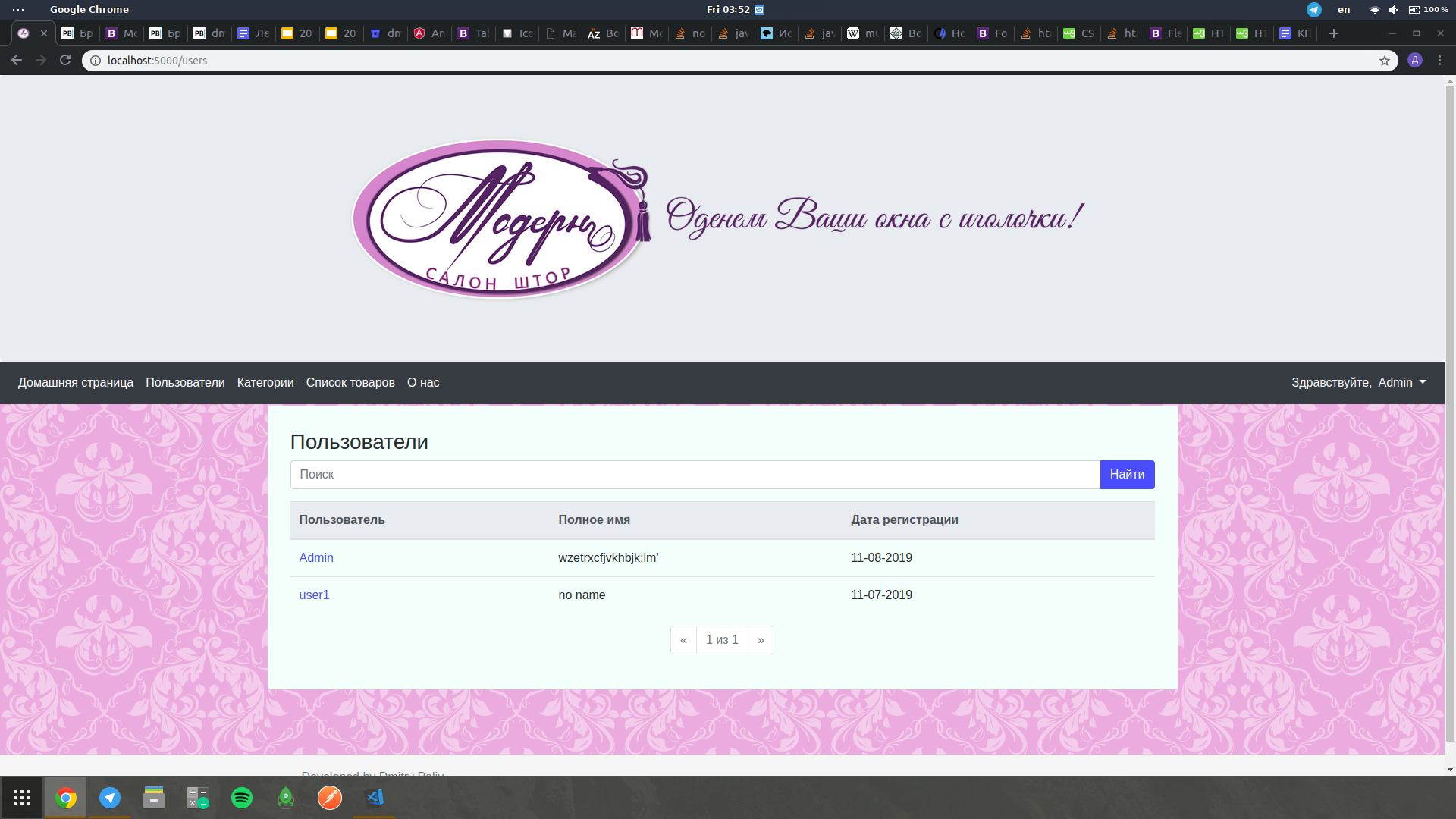
|  |
| --- |
| **products.js** |
| const entryLimit = 2;  let entryOffset = 0;  let entrySearchword = ''  Promise.all([  fetch(`/api/v1/products?limit=${entryLimit}&offset=${entryOffset}&searchword=${entrySearchword}`, { headers: { Authorization: `Bearer ${jwt}` }}).then(x => x.json()),  fetch('/templates/products.mst').then(x => x.text()),  fetch('/templates/pagination.mst').then(x => x.text())  ])  .then(([data, template, pagination]) => {  if (data.data.products) {  let pages = Math.floor(data.data.count / entryLimit);  if (data.data.count % entryLimit != 0) pages += 1;  const renderedHtmlStr = Mustache.render(template, data.data);  const renderedPagHtmlStr = Mustache.render(pagination, { pages: pages, page: 1 });  const prodEl = document.getElementById('productList');  prodEl.innerHTML = renderedHtmlStr;  const pagEl = document.getElementById('pagination');  pagEl.innerHTML = renderedPagHtmlStr;  if (entryOffset + entryLimit < data.data.count) $("#next").removeClass('disabled');  const nextEl = document.getElementById('next');  const prevEl = document.getElementById('prev');  const searchEl = document.getElementById('search');  nextEl.addEventListener('click', onNext);  prevEl.addEventListener('click', onPrev);  searchEl.addEventListener('click', onSearch);  const searchHelp = document.getElementById('searchHelp');  if (entrySearchword.length > 0) {  searchHelp.innerText = `Результаты поиска для: ${entrySearchword}`;  }  else searchHelp.innerText = '';  }  else {  const searchHelp = document.getElementById('searchHelp');  searchHelp.innerText = `Ничего не найдено`;  }  })  .catch(err => console.log(err));  function onNext(e) {  if (!e.target.className.includes('disabled')) {  entryOffset += entryLimit;  getPageData(entryLimit, entryOffset, entrySearchword);  }  }  function onPrev(e) {  if (!e.target.className.includes('disabled')) {  entryOffset -= entryLimit;  getPageData(entryLimit, entryOffset, entrySearchword);  }  }  function onSearch(e) {  entryOffset = 0;  const searchForm = document.getElementById('searchForm');  entrySearchword = searchForm.searchword.value;  getPageData(entryLimit, entryOffset, entrySearchword);  }  function getPageData(limit, offset, searchword) {  Promise.all([  fetch(`/api/v1/products?limit=${limit}&offset=${offset}&searchword=${searchword}`, { headers: { Authorization: `Bearer ${jwt}` }}).then(x => x.json()),  fetch('/templates/products.mst').then(x => x.text()),  ])  .then(([data, template]) => {  const renderedHtmlStr = Mustache.render(template, data.data);  const productList = document.getElementById('productList');  productList.innerHTML = renderedHtmlStr;  const pageNum = document.getElementById('pageNum');  let page = Math.floor(offset / limit) + 1;  let pages = Math.floor(data.data.count / entryLimit);  if (data.data.count % entryLimit != 0 || pages === 0) pages += 1;  pageNum.innerText = `${page} из ${pages}`;  if (offset + limit < data.data.count) $("#next").removeClass('disabled');  else $("#next").addClass('disabled');  if (offset != 0) $("#prev").removeClass('disabled');  else $("#prev").addClass('disabled');  const searchHelp = document.getElementById('searchHelp');  if (entrySearchword.length > 0) {  searchHelp.innerText = `Результаты поиска для: ${entrySearchword}`;  }  else searchHelp.innerText = '';  if (data.data.count === 0) {  const searchHelp = document.getElementById('searchHelp');  searchHelp.innerText = `Ничего не найдено`;  }  })  .catch(err => console.log(err));  } |

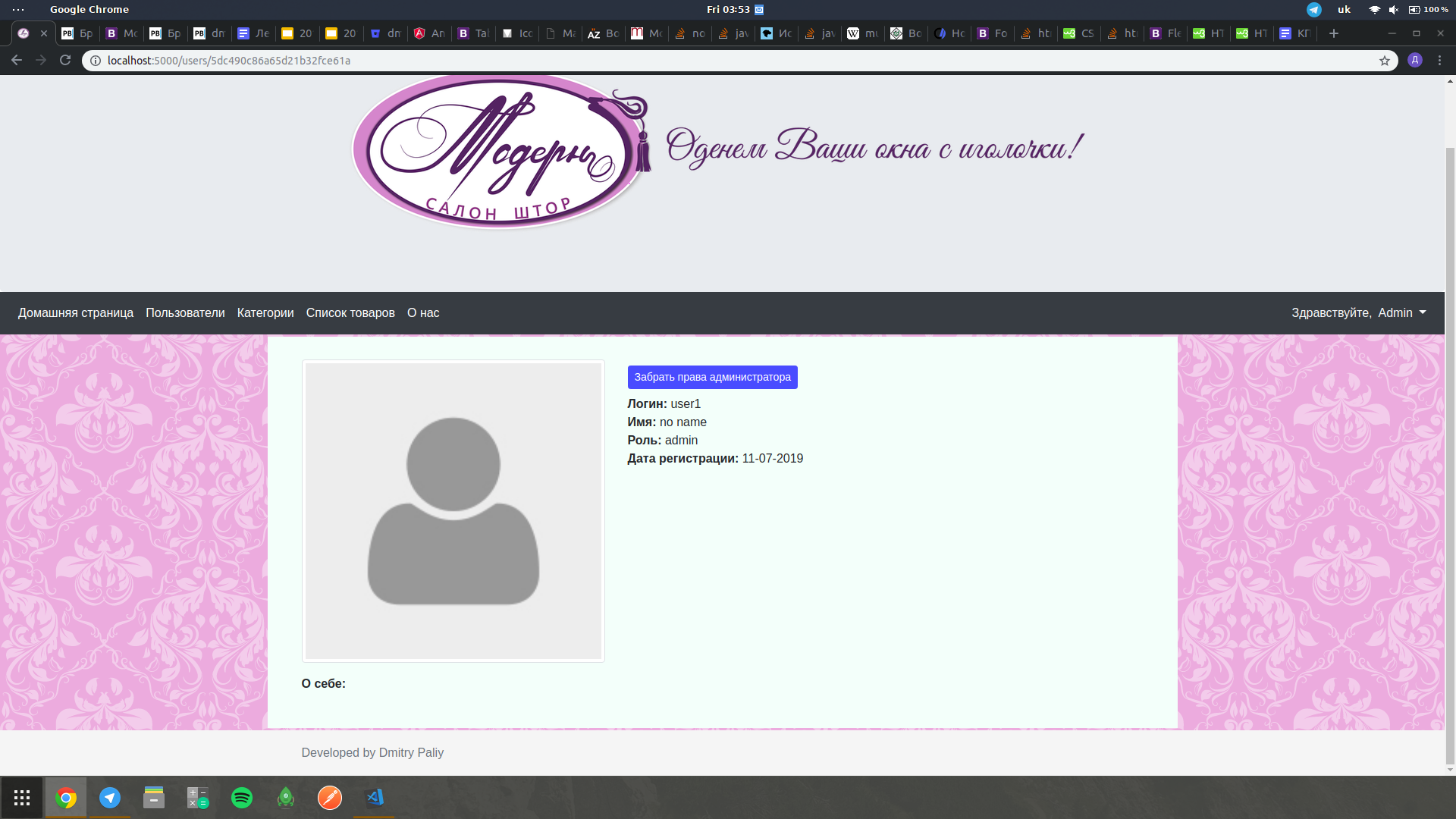
|  |
| --- |
| **product.js** |
| const url = document.location.href;  const id = url.split('/products/')[1];  let product = {};  Promise.all([  fetch(`/api/v1/products/${id}`, { headers: { Authorization: `Bearer ${jwt}` }}).then(res => {  if(res.status == 404) {  window.location.replace("/notfound");  return Promise.reject('Not Found');  }  else return res;  }).then(x => x.json()),  fetch('/templates/product.mst').then(x => x.text())  ])  .then(([data, template]) => {  if (data.data) {  product = data.data;  const renderedHtmlStr = Mustache.render(template, data.data);  const contents = document.getElementById('contents');  contents.innerHTML = renderedHtmlStr;  if (data.user.role == 'admin') {  $("#control").removeClass('d-none');  const deleteEl = document.getElementById('confirmDelete');  deleteEl.addEventListener('click', onDelete);  const editEl = document.getElementById('edit');  editEl.addEventListener('click', onEdit);  const confirmUpdate = document.getElementById('confirmUpdate');  confirmUpdate.addEventListener('click', onConfirmUpdate);  }  }  })  .catch(err => console.log(err));  window.onload = () => {  $('.custom-file-input').on('change',function() {  let fileName = $(this).val();  const chosenPhoto = document.getElementById('chosenPhoto');  chosenPhoto.innerHTML = fileName;  })  };  function onDelete(e) {  fetch(`/api/v1/products/${id}`, {  method: 'DELETE',  headers: {  'Authorization': `Bearer ${jwt}`  },  }).then(res => {  if(res.status == 404) {  window.location.replace("/notfound");  return Promise.reject('Not Found');  }  else return res;  })  .then((data) => {  if (data.err) Promise.reject(data.err);  else {  e.preventDefault();  window.location.replace("/products");  }  })  .catch(err => console.log(err));  }  function onEdit(e) {  document.getElementById('prodname').value = product.prodname;  document.getElementById('price').value = product.price;  document.getElementById('inStock').checked = product.inStock;  }  function onConfirmUpdate(e) {  const form = new FormData(document.getElementById('updateForm'));  fetch(`/api/v1/products/${id}`, {  method: 'PUT',  headers: {  'Authorization': `Bearer ${jwt}`,  },  body: form  }).then(res => {  if(res.status == 404) {  window.location.replace("/notfound");  return Promise.reject('Not Found');  }  else return res;  })  .then(data => {  document.location.reload(true);  })  .catch(err => console.log(err));  } |

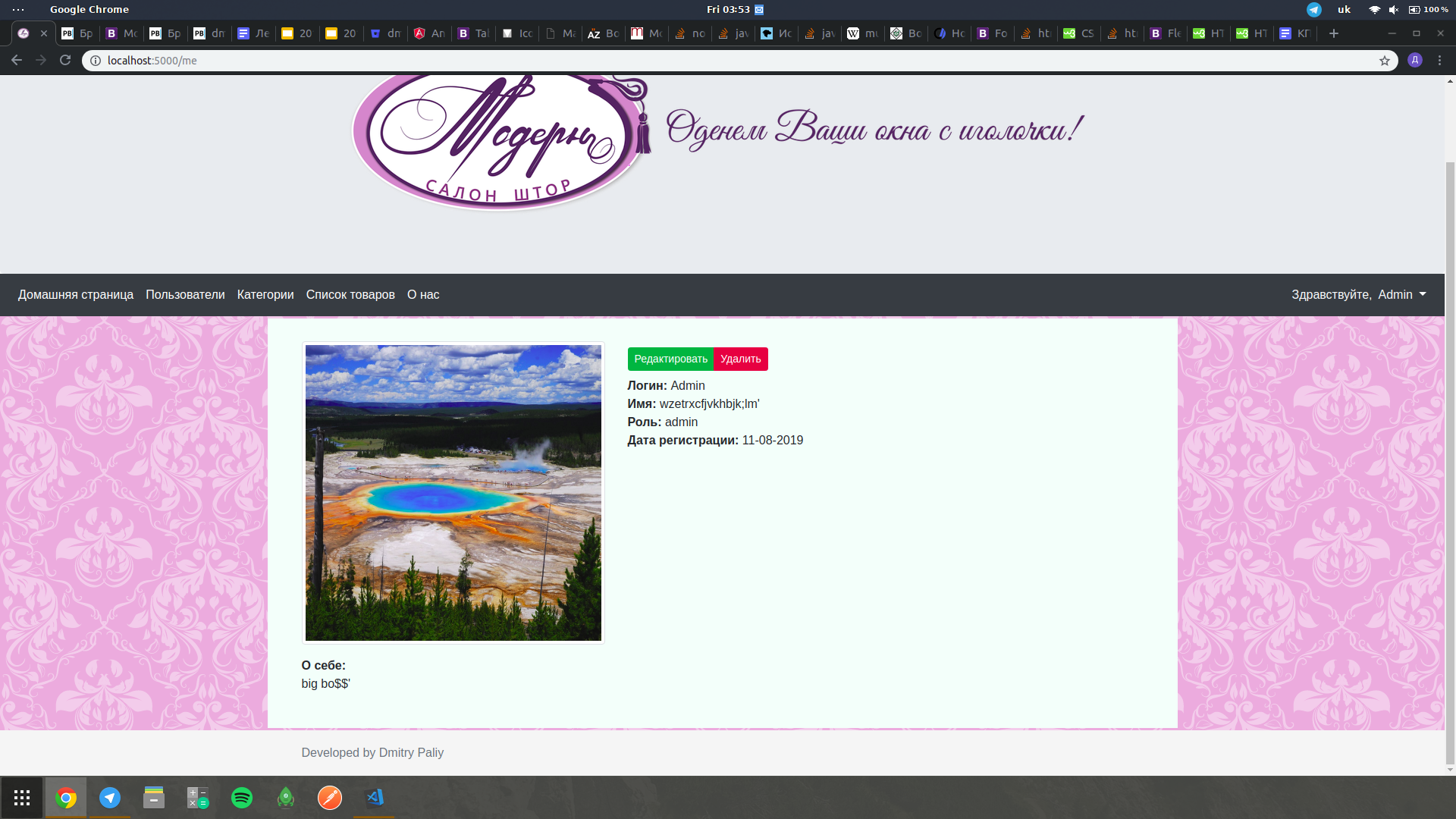
|  |
| --- |
| **profile.js** |
| let user = '';  window.onload = () => {  $('.custom-file-input').on('change',function() {  let fileName = $(this).val();  const chosenPhoto = document.getElementById('chosenPhoto');  chosenPhoto.innerHTML = fileName;  })  };  Promise.all([  fetch('/api/v1/me', { headers: { Authorization: `Bearer ${jwt}` }}).then(res => {  if(res.status == 404) {  window.location.replace("/notfound");  return Promise.reject('Not Found');  }  else return res;  }).then(x => x.json()),  fetch('/templates/user.mst').then(x => x.text())  ])  .then(([data, template]) => {  if (data.user) {  user = data.user;  const renderedHtmlStr = Mustache.render(template, data.user);  const contents = document.getElementById('contents');  contents.innerHTML = renderedHtmlStr;  $("#control").removeClass('d-none');  const deleteEl = document.getElementById('confirmDelete');  deleteEl.addEventListener('click', onDelete);  const editEl = document.getElementById('edit');  editEl.addEventListener('click', onEdit);  const confirmUpdate = document.getElementById('confirmUpdate');  confirmUpdate.addEventListener('click', onConfirmUpdate);  }  })  .catch(err => console.log(err));  function onDelete(e) {  fetch(`/api/v1/users/${user.\_id}`, {  method: 'DELETE',  headers: {  'Authorization': `Bearer ${jwt}`  },  }).then(res => {  if(res.status == 404) {  window.location.replace("/notfound");  return Promise.reject('Not Found');  }  else return res;  })  .then((data) => {  if (data.err) Promise.reject(data.err);  else {  e.preventDefault();  localStorage.removeItem('jwt');  window.location.replace("/");  }  })  .catch(err => console.log(err));  }  function onEdit(e) {  document.getElementById('fullname').value = user.fullname;  document.getElementById('bio').value = user.bio;  }  function onConfirmUpdate(e) {  const form = new FormData(document.getElementById('updateForm'));  fetch(`/api/v1/users/${user.\_id}`, {  method: 'PUT',  headers: {  'Authorization': `Bearer ${jwt}`  },  'Content-Type': 'multipart/form-data',  body: form  }).then(res => {  if(res.status == 404) {  window.location.replace("/notfound");  return Promise.reject('Not Found');  }  else return res;  })  .then(data => {  document.location.reload(true);  })  .catch(err => console.log(err));  } |

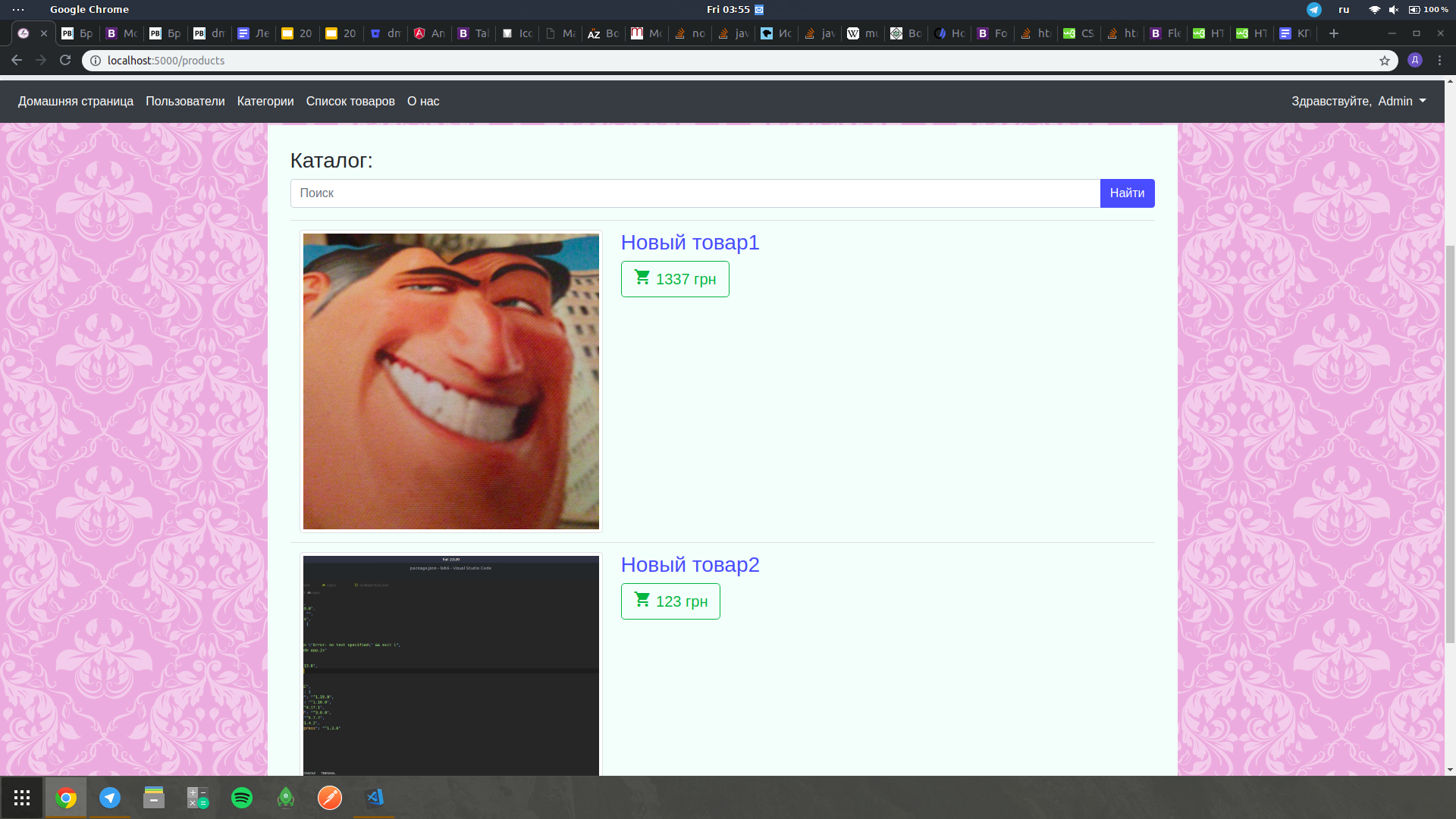
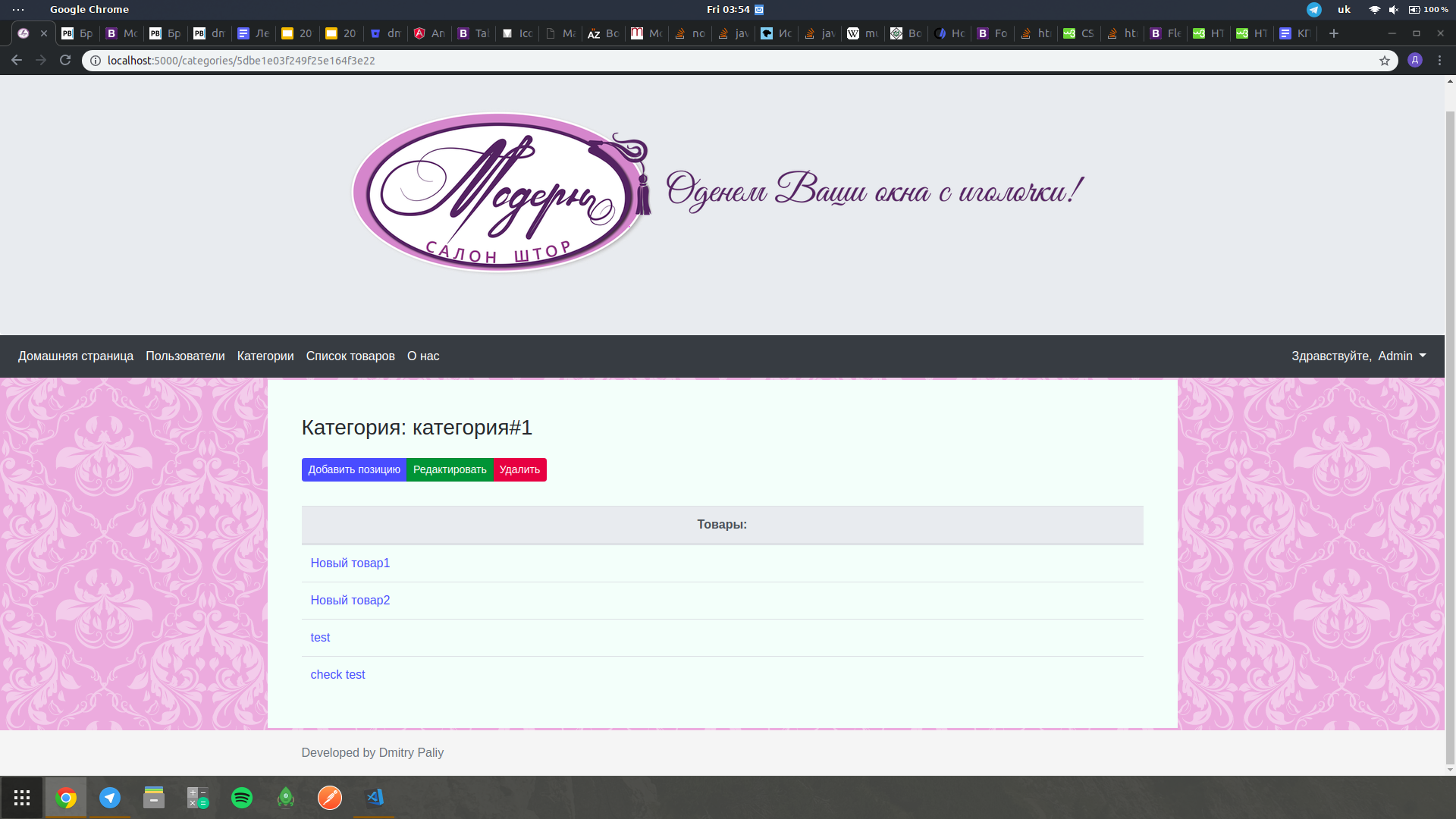
**Приклади результатів**

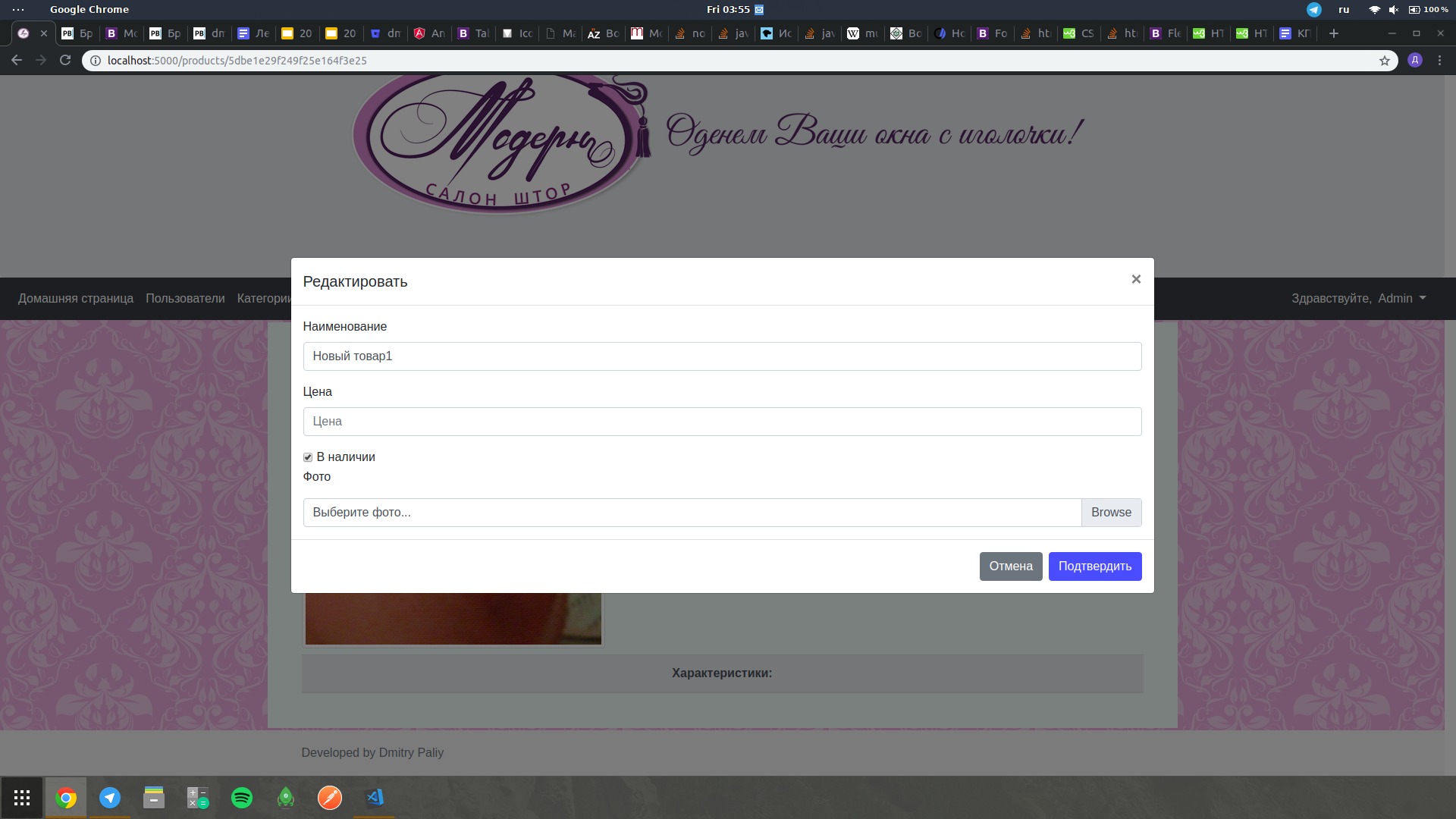
Скріншоти сайту:











**Висновки**

У цій лабораторній роботі я навчився застосовувати стилі Bootstrap, браузерний JS, ознайомився з технологією AJAX.