Sprawozdanie

Tabela: Kod – zdjęcie – laboratorium – temat – opis

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Kod | Zdjęcie | Laboratorium | Temat | Opis |
| function fetchRandomFact() {  // Load the articles  fetch('data/facts.json')  .then(response => response.json())  .then(data => {  // Parse the file and store the facts  const facts = data.facts;   // Generate a random index  const randomIndex = *Math*.floor(*Math*.random() \* facts.length);   // Retrieve the random fact  const randomFact = facts[randomIndex];   // Use the random fact as needed  *document*.getElementById('rand\_fact').innerHTML += `<p><h4>Ciekawy fakt!</h4></p><div class="fact">` + randomFact.content + `</div>`;   })  .catch(error => {  *console*.error('Error loading articles:', error);  }); } |  | Lab 12 | Praca z plikami .json w JavaScript | Kod pobiera dane z pliku .json i wybiera losowo jeden fakt i wypisuje go w rozdziale o Id ‘rand\_fact’, korzystując z .innerHTML |
| function OnClickInteractiveArticles(id) {  *document*.getElementById('main\_content').innerHTML = `<p><button onclick="OnClickInteractiveArticlesBack()" class="container">Do artykulów</button></p>`;   // Load the article  fetch('data/articles.json')  .then(response => response.json())  .then(data => {  // Parse the file and store the articles  const articles = data.articles;  const article = articles[id];  const name = article.title;  const content = article.content;  // Use the article as needed  *document*.getElementById('main\_content').innerHTML += `<div class="article container"><h3>` + name + `</h3><p>` + content + `</p></div>`;  })  .catch(error => {  *console*.error('Error loading articles:', error);  });   // Load the controls  *document*.getElementById('alt\_nav').innerHTML += `<button onclick="OnClickInteractiveArticlesBack()">Back</button>`; } |  | Lab 7, Lab 12 | Kod pobiera dane z pliku .json i tworzy stronę z artkulom wybranym przez użytkownika w rozdziale o Id ‘main\_content’, korzystując z .innerHTML.  Na stronie pojawia się przycisk do przywrócenia do listy artykulów |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| function OnClickInteractiveArticlesBack() {  *document*.getElementById('main\_content').innerHTML = `<h3>Dostępne artykuły:</h3>`;  fetchArticles(); } |  | Lab 7, Lab 12 | Praca z plikami .json w JavaScript | Wywołuje funkcje do przywrocenia początkowego wygłądu strony. Wyczyścza pole ‘main\_content’ korzystując z .innerHTML. |
| function fetchArticles() {  // Load the articles  fetch('data/articles.json')  .then(response => response.json())  .then(data => {  // Parse the file and store the articles  const articles = data.articles;   for (let i = 0; i < articles.length; i++) {  // Retrieve the article  const article = articles[i];  const preview = article.content.slice(0, 70) + '...';  const name = article.title.split(':')[0];  // Use the article as needed  *document*.getElementById('main\_content').innerHTML += `  <button class="container" onclick="OnClickInteractiveArticles(` + i + `)" ><h3>` + name + `</h3> <p>` + preview + `</p></button>  `;  }  })  .catch(error => {  *console*.error('Error loading articles:', error);  }); } | Kod pobiera dane z pliku .json i tworzy stronę z lisą artkulów w rozdziale o Id ‘main\_content’, korzystując z .innerHTML. |
| function validateForm(event) {  // Prevent form submission  event.preventDefault();   // Get form values  const name = *document*.getElementById("name").value;  const surname = *document*.getElementById("surname").value;  const email = *document*.getElementById("email").value;  const phone = *document*.getElementById("phone").value;  const birthdate = *document*.getElementById("birthdate").value;  const problemType = *document*.getElementById("problem-type").value;  const gender = *document*.querySelector('input[name="gender"]:checked');  const tag = *document*.querySelectorAll('input[name="tag"]:checked');  const about = *document*.getElementById("about").value;   // Perform additional validation if needed  const today = new *Date*();  const selectedDate = new *Date*(birthdate);  if (selectedDate > today) {  alert("Data urodzenia nie może być w przyszłości!");  return;  }   if (tag.length === 0) {  alert("Wybierz przynajmniej jeden tag!");  return;  }   // Save the form data to local storage  saveData(name, surname, email, phone, birthdate, problemType, gender, tag, about);   // Reset the form  event.target.reset();  displayData(); } | Brak | Lab 8, Żródło zewętrzne | Validacja formularzu przez JS | Kod pobiera dane z formularzu i sprawdza kalka pól przed tym jak zapisać dane w local storage. |
| function saveData(name, surname, email, phone, birthdate, problemType, gender, tag, about) {  // Get the existing form data from local storage  const existingFormData = *JSON*.parse(*localStorage*.getItem("formData"));   // Create an array with the new form data  const newFormData = [{  name,  surname,  email,  phone,  birthdate,  problemType,  gender: gender ? gender.value : null,  tag: *Array*.from(tag).map(input => input.value),  about  }];   // If existingFormData is an array, append the new form data to it  if (*Array*.isArray(existingFormData)) {  existingFormData.push(...newFormData);  } else {  // If existingFormData is not an array, create a new array with the new form data  *localStorage*.setItem("formData", *JSON*.stringify(newFormData));  return;  }   // Save the updated form data to local storage  *localStorage*.setItem("formData", *JSON*.stringify(existingFormData));  } | Brak | Lab 11, Lab12 | Zapis danych LocalStorage, praca z .json | Kod zapisuje podane parametrami dane jako listę i dodaje ich do localStorage |
| function displayData() {  // Get the form data from local storage  const formData = *JSON*.parse(*localStorage*.getItem("formData"));   // Check if there is any form data  if (!formData || formData.length === 0) {  *document*.getElementById("data-box").textContent = "No data available.";  return;  }   // Clear the previous content of the data box  *document*.getElementById("data-box").innerHTML = "";   // Create a container element for each form entry  formData.forEach((entry, index) => {  const container = *document*.createElement("div");  container.classList.add("entry-container");   // Create a heading with the entry number  const heading = *document*.createElement("h3");  heading.textContent = `Entry ${index + 1}`;  container.appendChild(heading);   // Create a paragraph for each form field and its value  for (const field in entry) {  const paragraph = *document*.createElement("p");  paragraph.innerHTML = `<strong>${field}:</strong> ${entry[field]}`;  container.appendChild(paragraph);  }   // Create edit and delete buttons  const editButton = *document*.createElement("button");  editButton.textContent = "Edit";  editButton.addEventListener("click", () => {  editEntry(index);  });  container.appendChild(editButton);   const deleteButton = *document*.createElement("button");  deleteButton.textContent = "Delete";  deleteButton.addEventListener("click", () => {  deleteEntry(index);  });  container.appendChild(deleteButton);   // Add the container to the data box  *document*.getElementById("data-box").appendChild(container);  }); } |  | Lab 11, Lab 12 | Odczyt dannych z localStorage, praca z .json | Kod odzytuje dane z localStorage i wypisuje ich do pola ‘data-box’  Dane są pogrupowane pod numerami ‘entry’.  Kod dodaje dwa przyciski do edycji i usunięcia zapisu |
| function editEntry(index) {  // Get the form data from local storage  const formData = *JSON*.parse(*localStorage*.getItem("formData"));   // Check if there is any form data  if (!formData || formData.length === 0) {  return;  }   // Retrieve the selected entry for editing  const selectedEntry = formData[index];   // Populate the form fields with the existing values  *document*.getElementById("name").value = selectedEntry.name;  *document*.getElementById("surname").value = selectedEntry.surname;  *document*.getElementById("email").value = selectedEntry.email;  *document*.getElementById("phone").value = selectedEntry.phone;  *document*.getElementById("birthdate").value = selectedEntry.birthdate;  *document*.getElementById("field-of-study").value = selectedEntry.fieldOfStudy;  const genderInputs = *document*.querySelectorAll('input[name="gender"]');  genderInputs.forEach((input) => {  if (input.value === selectedEntry.gender) {  input.checked = true;  }  });  const memeTypeInputs = *document*.querySelectorAll('input[name="memeType"]');  memeTypeInputs.forEach((input) => {  if (selectedEntry.memeTypes.includes(input.value)) {  input.checked = true;  }  });  *document*.getElementById("about").value = selectedEntry.about;   // Remove the selected entry from the form data array  formData.splice(index, 1);   // Save the updated form data to local storage  *localStorage*.setItem("formData", *JSON*.stringify(formData));   // Optionally, you can scroll to the top of the form for better visibility  *document*.documentElement.scrollTop = 0;   displayData(); } |  | Lab 11, Lab 12, Lab 4 | Praca z formularzem, Obsługa LocalStorage, .json, | Kod pobiera dane do zmiany i zapisuje ich u formularz dla szybkiego dostępu. Po wszystkich zmianach zapisuje w LocalStorage i usuwa stary rekord |
| function deleteEntry(index) {  // Get the form data from local storage  const formData = *JSON*.parse(*localStorage*.getItem("formData"));   // Check if there is any form data  if (!formData || formData.length === 0) {  return;  }   // Remove the entry from the form data array  formData.splice(index, 1);   // Save the updated form data to local storage  *localStorage*.setItem("formData", *JSON*.stringify(formData));   // Refresh the displayed data  displayData(); } | brak | Lab 11 | Usunięcie dannych z LocalStorage | Kod usuwa wybrany przez użytkownika zapis. |
| function deleteAllData() {  *localStorage*.removeItem("formData");  *document*.getElementById("data-box").innerHTML = "All data has been deleted."; } |  | Lab 11 | Usunięcie dannych z LocalStorage | Kod usuwa wszystkie zapisy w localStorage |