Weboldal dokumentáció

Web technológiák 1.

GEIAL331-BL

2024/2025/I.

**Készítette:**

Kopecskó Zsolt Bsc

Mérnökinformatikus

IVO46O

**Miskolc, 2025**

**KHN Dekor Weboldal**

**Bevezetés**

Ez a dokumentáció részletesen bemutatja a KHN Dekor weboldalának szerkezetét, működését, valamint a megvalósításhoz használt lényegesebb vezérlő elemeket és működésüket.

A weboldal egy kézműves termékeket forgalmazó egyéni vállalkozás termékeit bemutató oldal.

**1. Fájlstruktúra**

A projekt a következő fájlokat tartalmazza:

* **HTML fájlok**:
  + index.html: A főoldal, amely bemutatja a cég tevékenységeit és termékkategóriáit.
  + about.html: A vállalkozásról szóló információk és a csapat bemutatása.
  + contact.html: Kapcsolati információkat és térképet tartalmaz.
  + products.html: Termékeket bemutató oldal kategóriák szerint.
* **CSS fájlok**:
  + style.css: Az oldal megjelenéséért felelős stíluslap.
* **JavaScript fájlok**:
  + script.js: Az oldal interaktív elemeinek működését biztosítja.

**2. HTML Fájlok Részletes Elemzése**

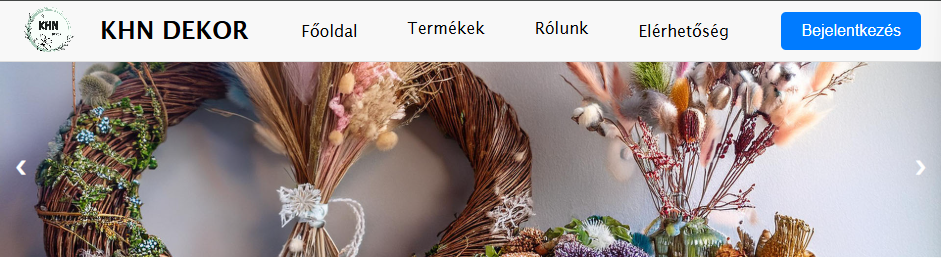
**2.1. index.html - Főoldal**

* **Navigációs menüsor**: Linkeket tartalmaz az összes fő oldalhoz, melyek felé az egérmutatót helyezve egy animált aláhúzás jelenik meg. A navigációs sáv tartalmaz továbbá egy logót, a vállalkozás nevét valamint egy bejelentkezés gombot.



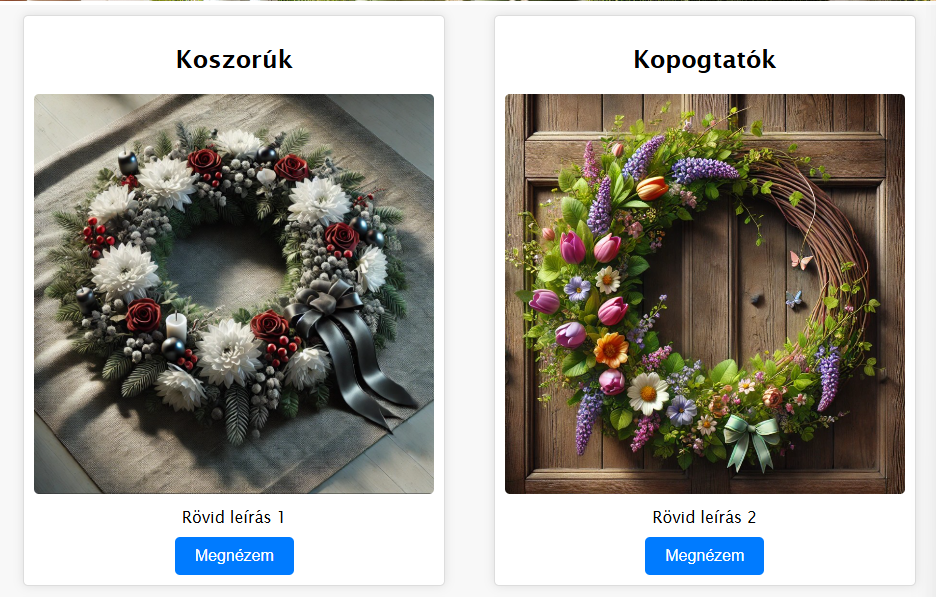
* **Slider**: Képek váltakozása az oldal fő részén.

A navigációs sáv alatt egy slider került elhelyezésre, mely három grafikát váltogat automatikusan 5 másodpercenként. A grafikák között lehetősége van a felhasználónak is váltani a képernyő szélén elhelyezett navigációs gombok segítségével.



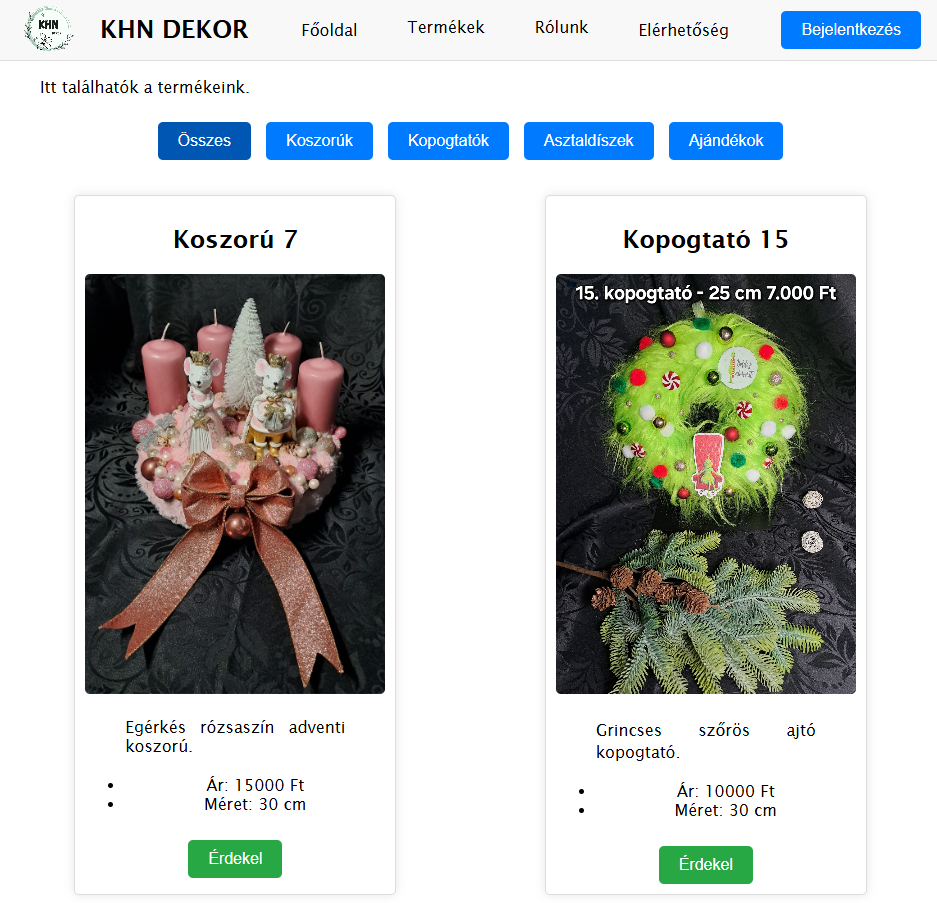
* **Termékkategóriák**: Gombok és képek segítségével kiemelt kategóriák bemutatása.

A weboldalon szereplő termékek 4égy fő kategóriába sorolhatóak. Ezeket a kategóriákat szemléltető boxok helyezkednek el a slider alatt. A boxok tartalmaznak egy gombot, melyre rákattintva az oldal átnavigál a termékek oldalra, ahol csak a kiválasztott kategóriához tartozó termékek jelennek meg.



**2.2. products.html - Termékek oldal**

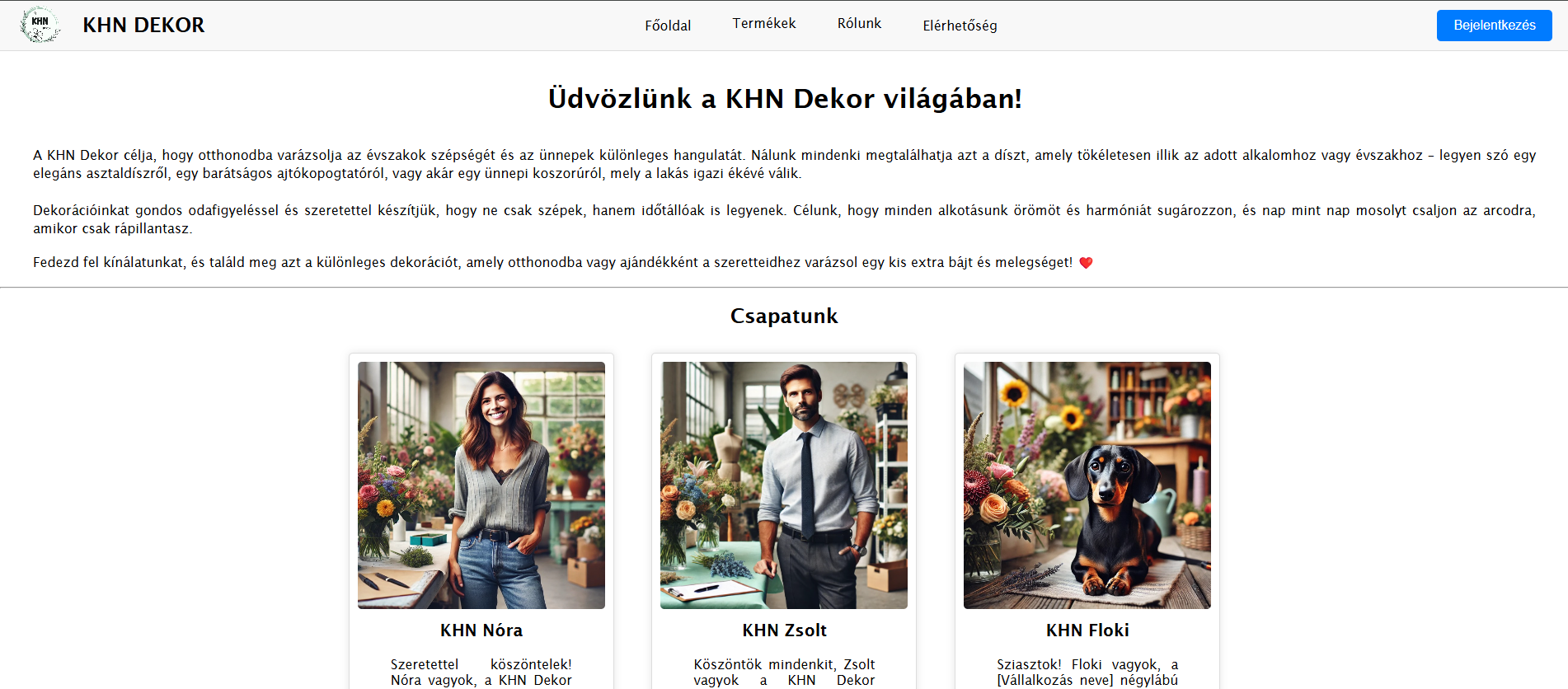
* **Kategóriák**: Különböző termékkategóriák szűrési lehetőségei.
* **Termékek listázása**: Egyedi képekkel, árakkal és leírásokkal.



A termékek oldalon alapértelmezetten az összes termék megjelenik külön-külön boxokban, melyek tartalmazzák a termék nevét, egy képet a termékről, egy rövid jellemző leírást, unordered list elemekként a termék vételárát és méretét, valamint egy érdekel gombot. a Gombra kattintva az oldal átnavigál a vállalkozás facebook oldalára.

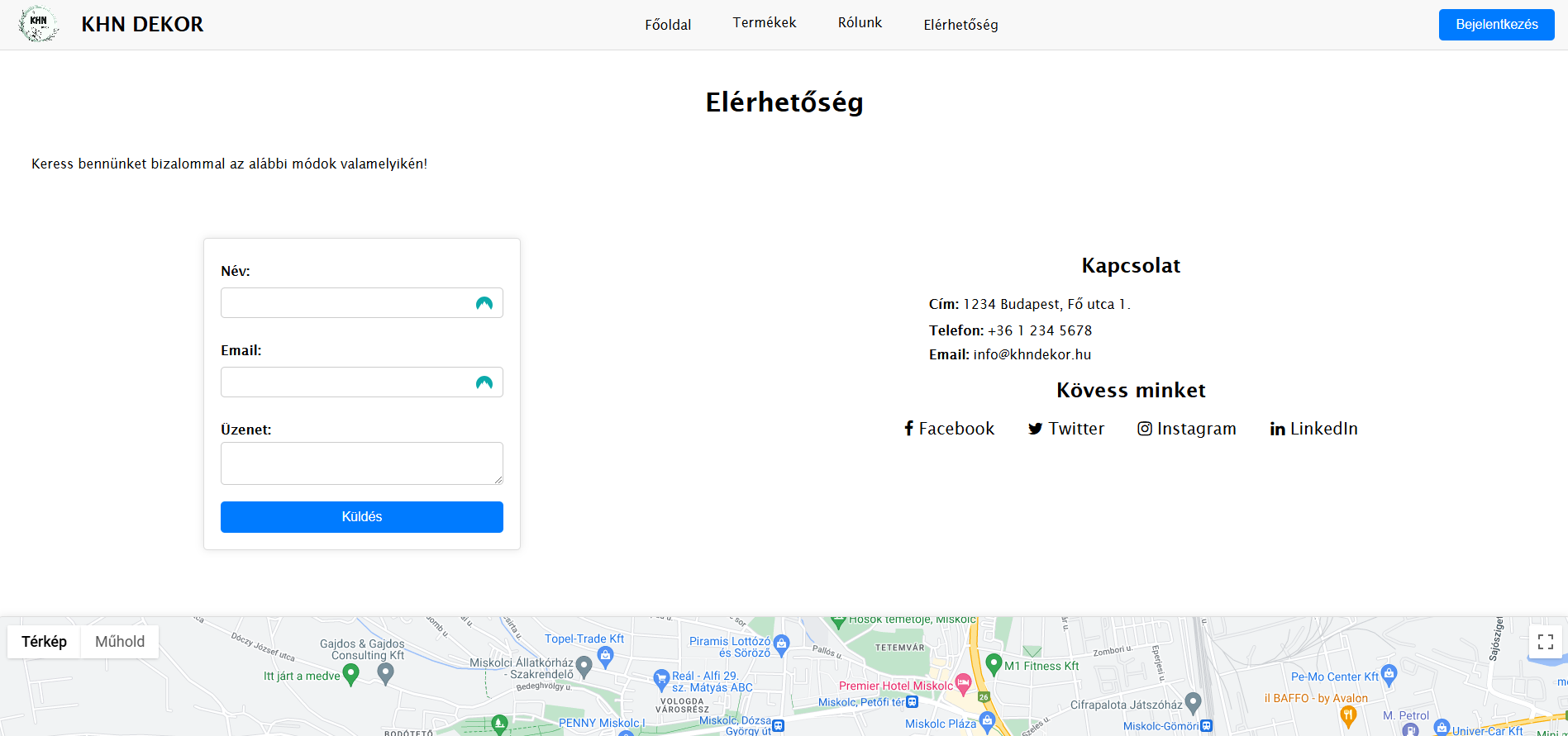
**2.3. about.html - Rólunk oldal**

* **Vállalkozás bemutatása**: Általános információk a KHN Dekor céljairól és értékeiről.
* **Csapat bemutatása**: A vállalkozás kulcsfontosságú szereplőinek ismertetése, képekkel és bemutatkozó szöveggel.



**2.4. contact.html - Kapcsolat oldal**

* **Kapcsolati információk**: Cím, telefonszám és email valamint social media linkek.
* **Kapcsolati űrlap**: Felhasználók üzeneteinek küldésére szolgáló mezők.
* **Térkép**: Google Maps integráció a vállalkozás helyszínének bemutatására.



**3. CSS Fájl - style.css**

**Stíluselemek:**

* A navigáció és a tartalom rugalmas elrendezéséhez a **felxbox** elrendezést használtam.
* **Színvilág**: Letisztult fekete, fehér és kék árnyalatok dominálnak.
* **Reszponzív dizájn**: Az oldal különböző képernyőméretekhez igazodó reszponzív dizájn kialakítását tartottam szemelőtt.
* **Az esztétikus le letisztult dizájn eléréshez** árnyékok, lekerekített sarkok valamint minimális animációk kerültek kialakításra.

**4. JavaScript Fájl - script.js**

**Funkciók:**

* **Slider vezérlése**: Automatikus és manuális képváltás funkciója.

**let slideIndex = 1;**

**showSlides(slideIndex);**

A diavetítés kezdeti beállítását végzi el. A slideIndex változó az aktuális diakép indexét tárolja, és kezdetben 1-re van állítva, ami azt jelenti, hogy az első diakép lesz az alapértelmezett megjelenített diakép.

A showSlides(slideIndex) függvényhívás felelős az átadott slideIndex paraméterrel a diaképek megjelenítéséért és elrejtéséért. A showSlides függvény végigmegy az összes diaképen, elrejti mindet, majd megjeleníti az aktuális diaképet, amelynek indexe megegyezik a slideIndex értékével.

// Diakép váltása navigációs gombokkal

**function plusSlides(n) {** // n: a diaképek számának növelése vagy csökkentésének paramétere, amely értéket a gomb kattintásakor kap 1 vagy -1 értéket vehet fel.

**showSlides(slideIndex += n);** // Az aktuális diakép indexének növelése vagy csökkentése

}

// Diaképek megjelenítése

**function showSlides(n) {** // n: a diakép indexe

// Az összes diakép kiválasztása

**let slides = document.querySelectorAll(".mySlides");**

// Ellenőrizd, hogy vannak-e diaképek, ha nincsenek, ne csinálj semmit

**if (slides.length === 0) return;**

// Ha a diakép index nagyobb, mint a diaképek száma, állítsd vissza az indexet 1-re

**if (n > slides.length) {**

**slideIndex = 1;**

**n = slideIndex;**

**}**

// Ha a diakép index kisebb, mint 1, állítsd az indexet a diaképek számának megfelelően

**if (n < 1) {**

**slideIndex = slides.length;**

**n = slides.length;**

**}**

// Az összes diakép elrejtése

**for (let i = 0; i < slides.length; i++) {**

**slides[i].style.display = "none";**

**}**

// Az aktuális diakép megjelenítése

**slides[slideIndex-1].style.display = "block";**

**}**

Az automatikus diakép váltás funkcióját valósítja meg. A startAutoSlide függvény egy belső függvényt, az autoSlide-ot definiál, amely a diaképek számának növeléséért felelős. Az autoSlide függvény a plusSlides(1) függvényt hívja meg, amely növeli az aktuális diakép indexét eggyel és megjeleníti a következő diaképet.

**function startAutoSlide() {**

**function autoSlide() {**

**plusSlides(1);**

**}**

**setInterval(autoSlide, 5000);**

**}**

A setInterval függvény segítségével az autoSlide függvény 5000 milliszekundumonként (azaz 5 másodpercenként) fut le. Így a diaképek automatikusan váltakoznak minden 5 másodpercben.

* **Modal ablakok kezelése**: Bejelentkezési és regisztrációs űrlapok megjelenítése.

**function initModal() {**

**const modal = document.getElementById('modal');**

**const loginBtn = document.querySelector('.login-btn');**

**const closeBtn = document.querySelector('.close-btn');**

**loginBtn.addEventListener('click', () => {**

**modal.style.display = 'block';**

**});**

**closeBtn.addEventListener('click', () => {**

**modal.style.display = 'none';**

**});**

**window.addEventListener('click', (event) => {**

**if (event.target == modal) {**

**modal.style.display = 'none';**

**}**

**});**

**}**

A fenti JavaScript kód a modális ablak inicializálásáért felelős. Az initModal függvény különböző HTML elemeket választ ki, amelyek a modális ablak megjelenítéséhez és bezárásához kapcsolódnak. A modal változó azonosítja a modális ablakot az id attribútuma alapján. A loginBtn és closeBtn változók azokat a gombokat vagy linkeket azonosítják, amelyekkel a felhasználó megnyithatja vagy bezárhatja a modális ablakot.

A loginBtn gombhoz egy kattintás eseményfigyelőt (click event listener) adunk hozzá. Amikor a felhasználó erre a gombra kattint, a modális ablak megjelenik (style.display = 'block'). Ez lehetővé teszi a felhasználó számára, hogy lássa és használja a modális ablak tartalmát.

Hasonlóképpen, a closeBtn gombhoz is hozzáadunk egy kattintás eseményfigyelőt. Amikor a felhasználó erre a gombra kattint, a modális ablak elrejtésre kerül (style.display = 'none'). Ez lehetővé teszi a felhasználó számára, hogy bezárja a modális ablakot, amikor már nincs rá szüksége.

Továbbá, egy eseményfigyelőt adunk az ablak (window) kattintás eseményéhez is. Ha a felhasználó a modális ablakon kívülre kattint, a modális ablak szintén elrejtésre kerül. Ez az eseményfigyelő ellenőrzi, hogy az esemény célpontja (event.target) megegyezik-e a modális ablakkal, és ha igen, akkor elrejti a modális ablakot.

**function initForms() {**

**const loginForm = document.getElementById('login-form');**

**const registerForm = document.getElementById('register-form');**

**const showRegister = document.getElementById('show-register');**

**const showLogin = document.getElementById('show-login');**

**showRegister.addEventListener('click', () => {**

**loginForm.style.display = 'none';**

**registerForm.style.display = 'block';**

**});**

**showLogin.addEventListener('click', () => {**

**registerForm.style.display = 'none';**

**loginForm.style.display = 'block';**

**});**

**}**

A fenti JavaScript kód az űrlapok inicializálásáért felelős. Az initForms függvény különböző HTML elemeket választ ki, amelyek a bejelentkezési és regisztrációs űrlapokhoz kapcsolódnak. A loginForm és registerForm változók azonosítják a bejelentkezési és regisztrációs űrlapokat az id attribútumuk alapján. A showRegister és showLogin változók azokat a gombokat vagy linkeket azonosítják, amelyekkel a felhasználó válthat a két űrlap között.

A showRegister gombhoz egy kattintás eseményfigyelőt (click event listener) adunk hozzá. Amikor a felhasználó erre a gombra kattint, a bejelentkezési űrlap (loginForm) elrejtésre kerül (style.display = 'none'), és a regisztrációs űrlap (registerForm) megjelenik (style.display = 'block'). Ez lehetővé teszi a felhasználó számára, hogy a regisztrációs űrlapot lássa és kitöltse.

Hasonlóképpen, a showLogin gombhoz is hozzáadunk egy kattintás eseményfigyelőt. Amikor a felhasználó erre a gombra kattint, a regisztrációs űrlap elrejtésre kerül, és a bejelentkezési űrlap jelenik meg. Ez lehetővé teszi a felhasználó számára, hogy visszaváltson a bejelentkezési űrlapra.

* **Kategória szűrés**: Termékek dinamikus megjelenítése a kiválasztott kategória alapján.

**function initCategoryButtons() {**

**const categoryButtons = document.querySelectorAll('.category-btn');**

**const productBoxes = document.querySelectorAll('.product-box');**

**categoryButtons.forEach(button => {**

**button.addEventListener('click', () => {**

**const category = button.getAttribute('data-category');**

**console.log(`Selected category: ${category}`); // Debug log**

**productBoxes.forEach(box => {**

**console.log(`Product box classes: ${box.classList}`); // Debug log**

**if (category === 'all' || box.classList.contains(category)) {**

**box.style.display = 'block';**

**} else {**

**box.style.display = 'none';**

**}**

**});**

**categoryButtons.forEach(btn => btn.classList.remove('active'));**

**button.classList.add('active');**

**});**

**});**

**}**

A fenti JavaScript kód a kategória gombok és termék dobozok kezeléséért felelős. Az initCategoryButtons függvény két fő elemet választ ki: a kategória gombokat (categoryButtons) és a termék dobozokat (productBoxes). A categoryButtons változó az összes .category-btn osztályú elemet tartalmazza, míg a productBoxes változó az összes .product-box osztályú elemet.

A kód minden egyes kategória gombhoz (categoryButtons) egy kattintás eseményfigyelőt (click event listener) ad hozzá. Amikor egy kategória gombra kattintanak, a kód lekéri a gombhoz tartozó kategóriát a data-category attribútumból, és ezt a kategóriát a konzolra írja ki a debug log segítségével.

Ezután a kód végigmegy az összes termék dobozon (productBoxes), és ellenőrzi, hogy a doboz tartalmazza-e a kiválasztott kategóriát. Ha a kategória all, vagy a doboz tartalmazza a kategóriát, akkor a doboz megjelenik (box.style.display = 'block'). Ellenkező esetben a doboz elrejtésre kerül (box.style.display = 'none'). A termék dobozok osztályait szintén a konzolra írja ki a debug log segítségével.

Végül a kód beállítja az aktív kategória gomb stílusát. Minden kategória gomb (categoryButtons) eltávolítja az active osztályt, majd a kattintott gombhoz hozzáadja az active osztályt, hogy vizuálisan jelezze a felhasználónak, melyik kategória van kiválasztva.

**function checkUrlParams() {**

**const urlParams = new URLSearchParams(window.location.search);**

**const categoryParam = urlParams.get('category');**

**if (categoryParam) {**

**const categoryButton = document.querySelector(`.category-btn[data-category="${categoryParam}"]`);**

**if (categoryButton) {**

**categoryButton.click();**

**}**

**}**

}A fenti JavaScript kód az URL paraméterek ellenőrzéséért és a kategória szűrés automatikus beállításáért felelős. A checkUrlParams függvény először létrehoz egy URLSearchParams objektumot a window.location.search értékéből, amely az aktuális URL keresési paramétereit tartalmazza. Ez az objektum lehetővé teszi az URL paraméterek egyszerű kezelését és lekérdezését.

A függvény ezután lekéri a category paraméter értékét az URL-ből a urlParams.get('category') hívással, és ezt az értéket a categoryParam változóban tárolja. Ha a categoryParam értéke nem null vagy undefined, azaz létezik ilyen paraméter az URL-ben, akkor a kód megkeresi a megfelelő kategória gombot a data-category attribútum alapján. Ezt a document.querySelector függvény segítségével teszi, amely visszaadja az első olyan elemet, amely megfelel a megadott CSS szelektornak.

Ha a megfelelő kategória gomb megtalálható (categoryButton nem null), akkor a kód szimulál egy kattintást a gombra a categoryButton.click() hívással. Ez automatikusan beállítja a kategória szűrést az oldal betöltésekor, mintha a felhasználó manuálisan kattintott volna a gombra.