**Střední průmyslová škola Ostrov, příspěvková organizace**

**ROČNÍKOVÁ PRÁCE**

***Katalog horských kol***

|  |  |  |
| --- | --- | --- |
| Studijní obor | **18-20-M / 01 - Informační technologie** | |
| Předmět: | **PGV – Programové vybavení**  **PVA – Programování a vývoj aplikací** | |
| Třída | **I3** | **David Topilnyckyj** |
| Školní rok | **2024 / 2025** | Jméno a příjmení autora |
|  |  | **Ing. Roman Stark**  **Mgr. Zuzana Hagarová** |
| Jméno a příjmení vedoucího práce |

„**Prohlašuji, že jsem tuto práci vypracoval(a) samostatně** a použil(a) jsem literárních pramenů a informací, které cituji a uvádím v seznamu použité literatury a zdrojů informací.“

Licenční ujednání:

1. Ve smyslu § 60 autorského zákona č. 121 / 2000 Sb. Poskytuji SPŠ Ostrov, Klínovecká 1197, 36301 Ostrov výhradní a neomezená práva (§ 46 a § 47) k využití mé ročníkové práce.

2. Bez svolení školy se zdržím jakéhokoliv komerčního využití této práce.

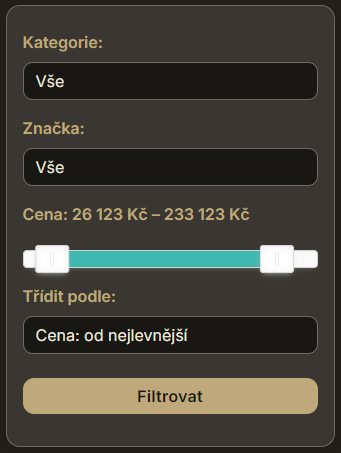
3. V případě komerčního využití práce školou obdrží žák – autor práce ( autorský kolektiv ) odměnu ve výši jedné třetiny dosaženého zisku.

4. Pro výukové účely a prezentaci školy se vzdávám nároku na odměnu za užití díla.

V Ostrově, dne ........................ …….........................

podpis

# Přílohy



Obrázek 1Vzhled Filtrů a třídění



Obrázek 2Bootsrap Card na webu



Obrázek 3 Detail vybraného kola

# Anotace práce – česky

Tento projekt byl vytvořen jako ročníková práce z předmětu Programování a vývoj aplikací. Cílem bylo navrhnout a následně realizovat jednoduchý webový katalog horských kol, který umožňuje uživatelům procházet nabídku horských kol, může si je filtrovat podle ceny, kategorie kola a podle značky kola. Následně si může zobrazit detaily kola o kterém má zájem se dozvědět více. Aplikace je rozdělená na dvě části. Uživatelská část, kde si návštěvníci prochází a zobrazují kola dle zadaných filtrů. Administrátorská část, která po přihlášení umožňuje přidávání, upravování a mazání záznamu v katalogu

V projektu se využívá PHP, MYSQL a Bootstrap 5 a knihovnu noUiSlider pro cenový filtr. Uživatelské rozhraní je přizpůsobeno jak pro desktop, tak i pro mobilní telefony.

# Anotace práce – anglicky nebo německy

This project was created from the subject Programming and Application Development. The aim was to design and then implement a simple web-based mountain bike catalogue that allows users to browse a range of mountain bikes, filter by price, bike category and bike brand. He can then view the details of the bike he is interested in learning more about. The application is divided into two parts. The user part, where visitors browse and view bikes according to the filters thee specify. The admin part, which after login allows adding, editing and deleting a catalog entry

The project uses PHP, MYSQL and Bootstrap 5 and the noUiSlider library for the price filter. The user interface is customized for both desktop and mobile.

# Seznam použitých značek a symbolů

ICT Informační a komunikační technologie

PHP

MYSQL

noUiSlider

# Obsah

[Přílohy 3](#_Toc199839356)

[Anotace práce – česky 5](#_Toc199839357)

[Anotace práce – anglicky nebo německy 5](#_Toc199839358)

[Seznam použitých značek a symbolů 6](#_Toc199839359)

[Obsah 7](#_Toc199839360)

[1. Úvod 1](#_Toc199839361)

[2. Databáze a rozhraní 3](#_Toc199839362)

[2.1. Vstupní data MySQL 3](#_Toc199839363)

[2.2 Zobrazení v uživatelské části 6](#_Toc199839364)

[2.3 Hlavní stránka 7](#_Toc199839365)

[2.3.2 access.php 7](#_Toc199839366)

[2.3.2 Spojení s databází 16](#_Toc199839367)

[2.3.3 Design hlavní stránky 17](#_Toc199839368)

[3. Detail horského kola 31](#_Toc199839369)

[3.1. Načtení dat z databáze 31](#_Toc199839370)

[3.2. Vzhled a rozvržení 32](#_Toc199839371)

[4. Administrátor 35](#_Toc199839372)

[3 Závěr 37](#_Toc199839373)

[Citovaná literatura 38](#_Toc199839374)

[Seznam obrázků 39](#_Toc199839375)

[Seznam tabulek 39](#_Toc199839376)

[Seznam grafů 39](#_Toc199839377)

[Seznam příloh 39](#_Toc199839378)

[Seznam použitého softwaru 40](#_Toc199839379)

[Seznam použitých odborných výrazů 41](#_Toc199839380)

## Úvod

Tato ročníková práce vznikla jako závěrečný projekt v rámci předmětu Programování a vývoj aplikací. Jejím hlavním cílem bylo navrhnout a vytvořit funkční a moderně vypadající webový katalog horských kol, který by byl srozumitelný, přívětivý k uživatelům a umožňoval rychlé a efektivní vyhledávání a filtrování produktů podle různých parametrů – například podle ceny nebo konkrétní kategorie.

V práci se podrobně věnuji celému procesu tvorby webového katalogu – od úvodní analýzy požadavků a potřeb, přes návrh databázové struktury, až po samotné naprogramování uživatelského rozhraní a jednotlivých funkcí. Zásadní snahou bylo vytvořit katalog, který nabídne kompletní přehled všech dostupných horských kol, umožní jejich filtrování dle značky, typu či cenové hladiny a zároveň bude přehledný a jednoduše ovladatelný, a to i na mobilních zařízeních.

Rozsah projektu byl omezen na vytvoření funkčního prototypu, který ukazuje jak základní, tak i některé pokročilejší funkce, jež by měl moderní webový katalog umět. Myslel jsem i do budoucna – návrh počítá s tím, že katalog bude možné dále rozšiřovat, například o možnost přímé objednávky, uživatelské recenze nebo doplnění podrobnějších technických specifikací, čímž by se z něj mohl stát plnohodnotný e-shop.

Samotný projekt jsem rozdělil do několika hlavních etap. Nejprve jsem si připravil návrh toho, jak by měl katalog vypadat a jaké klíčové funkce by měl obsahovat. Poté následoval návrh databáze, která uchovává všechny důležité informace o jednotlivých kolech – od značky a modelu, přes popis a cenu, až po zařazení do konkrétní kategorie. Následně jsem se věnoval návrhu a vývoji uživatelského rozhraní pomocí HTML, CSS a JavaScriptu. Toto rozhraní slouží k přehlednému zobrazení nabízených produktů a umožňuje jejich filtrování podle zvolených kritérií. Komunikaci s databází jsem zajistil pomocí PHP a MySQL – technologií, které se běžně používají při vývoji webových aplikací. Během celé tvorby jsem jednotlivé části průběžně testoval a upravoval podle zjištěných potřeb. Klíčovým požadavkem bylo, aby byl katalog plně responzivní, tedy aby dobře fungoval i na menších obrazovkách, jako jsou tablety nebo chytré telefony, což s sebou přinášelo i určité výzvy.

V následujících kapitolách této práce se podrobně zaměřím na jednotlivé kroky tvorby katalogu. Nejprve popíšu analýzu požadavků a potřeb, pak návrh databázového modelu a jeho realizaci. Následovat bude část věnovaná vývoji uživatelského rozhraní, dále vysvětlím principy filtrování a závěrem přiblížím, jak jsem zajistil responzivní vzhled aplikace. V poslední části se budu věnovat zhodnocení celého projektu – co se mi povedlo, kde vidím prostor pro vylepšení a jaké zkušenosti jsem při práci získal.

Úvodní část této práce shrnuje, proč jsem si pro projekt zvolil právě téma katalogu horských kol, jaký byl jeho hlavní záměr a jak jsem postupoval při jeho realizaci. Celá práce je zaměřená na praktickou stránku vývoje webové aplikace, jejímž cílem je srozumitelné zobrazení a efektivní filtrování produktů. V dalších částech se podrobněji věnuji jednotlivým fázím vývoje a prezentuji výsledky svého ročníkového projektu.

## Databáze a rozhraní

### Vstupní data MySQL

Při navrhování a následném vytváření ročníkové práce bylo velmi důležité si správně rozmyslet jak a jaká data půjdou do jakých tabulek, aby v tom byl pořádek a mělo to nějakou strukturu ve které by se vyznal každý. Když máte dobře rozvržená data tak to usnadňuje mnohonásobně práci, což jsem si i sám vyzkoušel. Nedokážu spočítat kolikrát jsem svoje tabulky předělával přidával nové a mazal ty špatné. Mojí databázi tvoří celkem 4 tabulky. V této části popíšu strukturu každé z nich, včetně jejich vzájemných vazeb.

#### Tabulka bikes

Tato tabulka je klíčová pro tuto aplikaci. Uchovávám zde všechny informace o jednotlivých kolech, která se zobrazují uživatelům při prohlížení webu.

A screenshot of a computer

AI-generated content may be incorrect.

Obrázek 4 Tabulka bikes v admineru

Jak jsem již zmiňoval tabulka je klíčová pro celou aplikaci. Jsou zde všechny informace o jednotlivých kolech, která se zobrazují uživatelům. Každý záznam obsahuje:

* Id – unikátní identifikátor kola (je automaticky generovaný), to že na obrázku je to tam tak že pár čísel chybí, protože jsem ty kola vymazal, na funkci nevadí
* Model a color – název kola a barva daného kola
* Description – popis kola, z oficiálních stránek prodejců, co bylo anglicky jsem přeložil pomocí deepl.com
* Price – cena daného kola, cena se brala z hlavních stránek prodejce (kurz eura 24,90)
* Image\_main – název hlavního obrázku, (obrázky jsou uložené ve složce img/ tak aby se daly přečíst a vypsat, musí sedět pojmenování obrázku)
* Image\_gallery – názvy dalších obrázků (uložené jako řetězec oddělený čárkami)
* Category\_id – cizí klíč který odkazuje na další tabulku, a to konkrétně na tabulku kategorií
* Brand\_id – cizí klíč odkazující na další tabulku, konkrétně tabulku značek brands o které budu psát níže.

Pomocí těchto sloupců můžu každé kolo přiřadit ke konkrétní kategorii i značce a zároveň zobrazit všechny důležité informace v uživatelském rozhraní.

#### Tabulka categories

Tabulku categories jsem vytvořil pro uchovávaní typů kol, aby bylo možné je efektivně filtrovat. Když si to takhle roztahám po tabulkách tak se v tom lépe vyznám. Každá kategorie má svůj daný typ kola, od kterého se odvíjí specifikace, volba komponent a taky cena.

A screenshot of a computer

AI-generated content may be incorrect.

Obrázek 5 tabulka categories v admineru

* Id – primární klíč
* Category – název kategorie (XC, Trail, Enduro a Downhill) toto jsou kategorie kol

Tuto tabulku využívám při zadávání nového kola, kdy zvolím kategorii, do které patří dané přidávané kolo, pomocí výběru. Ve výsledném katalogu pak mohu nabídnout uživatelům možnost filtrovat aktuální kola podle typu který je zajímá.

#### Tabulka brands

Stejným způsobem jako tabulka categories jsem navrhl i správu značek kol. Tabulka brands obsahuje značky kol, která vyrábí horská kola.

A table with text on it

AI-generated content may be incorrect.

Obrázek 6 tabulka brands v admineru

* Id – primární klíč
* Brand název značky (Trek, Specialized, YT, Giant, Norco, Canyon a další)

Tuto tabulku využívám při vkládaní i filtrování kol podle značek. Umožňuje mi spravovat značky centrálně bez nutnosti je znovu vypisovat u každého kola.

#### Tabulka admins

Tabulku admins jsem navrhl kvůli zabezpečení administrativního rozhraní. Ukádám zde přihlašovací údaje administrátora, jméno a heslo.

* Id – primární klíč
* Username – jméno administrátora, zde jsem nastavil username a admin
* Password – heslo správce (uložené ve formátu varchar, v budoucnu doporučuju hashování)

Tyto údaje porovnávam při přihlášení administrátora na web stránce a následném přístupu do rozhraní pro správu kol. Zde administrátor může přidávat kola, mazat kola, upravovat různé detaily.

**Vazby mezi tabulkami**

Databázové tabulky spolu úzce souvisí:

V tabulce bikes odkazuji pomocí sloupců category\_id a brand\_id na tabulky categorie a brands. Díky tomu mohu efektivně získávat názvy kategorií a značek pro každé kolo.

Tabulka admins je oddělená od ostatních a slouží pouze pro ověření oprávněného přístupu ke správě obsahu. Její funke je bezpečnostní a nesouvisí s obsahem katalogu.

Tento návrh databázové struktury mi umožnil vytvořit systém, který je přehledný, rozšiřitelný a snadno spravovatelný z administrátorské části. Jednotlivé tabulky jsem navrh tak, aby bylo možné jednoduše doplňovat nové záznamy a zároveň zachovat logické propojení mezi kategoriemi, značkami a konkrétními produkty.

### Zobrazení v uživatelské části

Uživatelská část katalogu horských kol představuje rozhraní, které je určeno všem návštěvníkům webu bez přihlášení. Klade důraz na jednoduchost, přehlednost a moderní design v černo – zlaté barevné kombinaci. Hlavní stránka (access.php) obsahuje všechny důležité prvky pro výběr a prohlížení dostupných kol.

Po načtení stránky uživatel vidí následující části: Hlavní nadpis, pod ním filtry na levé straně spolu s tříděním dat, který umožňuje filtrovat podle: kategorie (generované z databáze categories). Mám zde filtr pro cenu jako slider. Dále pak filtrace podle značky kola. Dále pak třídění dat podle ceny: od nejdražšího kola po nejlevnější a od nejlevnějšího po nejdražší, a třídění dat podle Modelu: od A do Z a naopak.

Seznam všech dostupných kol, který se zobrazuje ve formě karet (cards) pomocí bootstrap 5. Každá karta obsahuje obrázek kola, název kola, jeho cenu a tlačítko pro zobrazení detailu.

Web je navržen s důrazem na responzivní zobrazení – obsah se přizpůsobuje různým velikostem obrazovek, takže katalog je přehledný jak na počítači. K dosažení moderního vzhledu byl použit vlastní soubor style.css, kde jsou definovány například:

* barvy (černobílá kombinace, decentní přechody),
* ohraničení a zaoblení prvků (např.… card s border-radius),
* stylování filtru a tlačítek.

Po kliknutí na tlačítko zobrazit detail se uživatel přesune na stránku detail.php, kde se zobrazí: název kola, hlavní obrázek kola, doplňková fotogalerie, popis kola, kategorie a cena. Díky jednoduchému Javascriptovému scriptu se doplňkové obrázky dynamicky přepínají, čímž vytváří efekt interaktivní fotogalerie. Celý katalog je navržen tak, aby uživatel mohl rychle a pohodlně najít kolo podle svých požadavků. Použití databáze umožňuje snadné rozšíření o další modely kol, značky i kategorie bez zásahu do kódu.

### Hlavní stránka

### access.php

Soubor access.php představuje hlavní vstupní bod uživatelské části webové aplikace – katalogu horských kol. Na této stránce se zobrazují všechny veřejně přístupné informace o nabízených kolech, uživatel má možnost filtrovat záznamy podle kategorie a ceny a kliknout na detailní náhled vybraného kola. Kromě toho se na této stránce nachází i modální přihlašovací formulář pro administrátory, kteří následně přecházejí do administrační části aplikace, ale o tom budu mluvit níže.

Celkově je soubor rozdělen do několika funkčních sekcí, které spolu tvoří jednotnou a responzivní uživatelskou část katalogu. Kód využívá kombinaci jazyka PHP pro práci s databází a dynamické generování obsahu, HTML pro strukturu stránky, CSS (přes Bootstrap pro stylování a JavaScript (včetně jQuery) pro zajištění interaktivity.

Začneme rozepsáním php části této stránky a postupně ji budu popisovat.

Soubor začíná tím, že naimportuje nastavení pro databázové připojení. Tento soubor obsahuje připojení k MySQL přes PDO, které nabízí vyšší bezpečnost oproti klasickému mysqli\_connect(). Aby bylo možné správně nastavit posuvník ceny, aplikace dotazuje databázi na minimální a maximální hodnotu cen. Kromě cenového filtru umožňuje stránka uživateli filtrovat seznam kol podle kategorie (např. MTB, silniční kola) a značky (např. Trek, Specialized, Giant). Každý záznam obsahuje ID a název dané značky/kategorie. Data jsou následně použita k vykreslení dynamických <select> filtrů v HTML formuláři. Abychom zajistili bezpečné a správné filtrování, zpracováváme vstupní parametry. Použitím (int) eliminujeme riziko SQL injection a jiných zneužití. Na základě uživatelských filtrů sestavujeme parametrizovaný SQL dotaz, který zabrání bezpečnostním hrozbám a zajistí správné výsledky. Uživatel může zvolit způsob řazení položek pomocí sortOption, které upravuje SQL dotaz. Stránka používá Bootstrap 5 pro stylování a responzivní design. Hlavní části zahrnují:

* Navigační lištu s tlačítkem pro přihlášení administrátora.
* Sidebar s filtry, který se dynamicky skrývá na mobilních zařízeních.
* Výpis kol v kartách (<div class="card">).
* Modální přihlašovací formulář pro administrátory.

<?php

require 'database.php';

// Získání minimální a maximální ceny z databáze

$priceQuery = $pdo->query("SELECT MIN(price) AS min\_price, MAX(price) AS max\_price FROM bikes");

$priceRange = $priceQuery->fetch(PDO::FETCH\_ASSOC);

$minPrice = (int)$priceRange['min\_price'];

$maxPrice = (int)$priceRange['max\_price'];

// Načtení značek a kategorií

$brandStmt = $pdo->query("SELECT \* FROM brands");

$brands = $brandStmt->fetchAll(PDO::FETCH\_ASSOC);

$categoryStmt = $pdo->query("SELECT \* FROM categories");

$categories = $categoryStmt->fetchAll(PDO::FETCH\_ASSOC);

// Vybrané filtry

$selectedCategory = isset($\_GET['category']) ? (int) $\_GET['category'] : 0;

$selectedBrand = isset($\_GET['brand']) ? (int) $\_GET['brand'] : 0;

$filterMin = isset($\_GET['min\_price']) ? (int) $\_GET['min\_price'] : $minPrice;

$filterMax = isset($\_GET['max\_price']) ? (int) $\_GET['max\_price'] : $maxPrice;

$sortOption = isset($\_GET['sort']) ? $\_GET['sort'] : 'price\_asc';

$params = [

    'min\_price' => $filterMin,

    'max\_price' => $filterMax

];

$sql = "SELECT bikes.id, bikes.model, bikes.color, bikes.price, bikes.image\_main,

        brands.Brand AS brand\_name, categories.Category AS category\_name

        FROM bikes

        JOIN brands ON bikes.brand\_id = brands.id

        JOIN categories ON bikes.category\_id = categories.id

        WHERE bikes.price BETWEEN :min\_price AND :max\_price";

if ($selectedCategory > 0) {

    $sql .= " AND bikes.category\_id = :category\_id";

    $params['category\_id'] = $selectedCategory;

}

if ($selectedBrand > 0) {

    $sql .= " AND bikes.brand\_id = :brand\_id";

    $params['brand\_id'] = $selectedBrand;

}

switch ($sortOption) {

    case 'price\_asc':

        $sql .= " ORDER BY bikes.price ASC";

        break;

    case 'price\_desc':

        $sql .= " ORDER BY bikes.price DESC";

        break;

    case 'az':

        $sql .= " ORDER BY bikes.model ASC";

        break;

    case 'za':

        $sql .= " ORDER BY bikes.model DESC";

        break;

    default:

        $sql .= " ORDER BY bikes.price ASC";

}

$stmt = $pdo->prepare($sql);

$stmt->execute($params);

$bikes = $stmt->fetchAll(PDO::FETCH\_ASSOC);

?>

Nyní se přesouváme do Javascriptové části kodu. Soubor access.php obsahuje několik interaktivních prvků, které zvyšují uživatelskou přívětivost a efektivitu filtrování. Tyto prvky zahrnují:

* Dynamický cenový posuvník (noUiSlider), který umožňuje plynulé nastavení cenového rozpětí.
* Hover efekt na obrázky kol, který vytváří vizuálně atraktivní zvětšení obrázků při najetí myší.
* Automatická synchronizace filtrů, kdy se zadané hodnoty aktualizují v formulářových polích, což usnadňuje jejich odesílání.

Cena je jedním z hlavních parametrů, který uživatelé často filtrují. Namísto standardních textových polí jsme použili **noUiSlider**, což je JavaScriptová knihovna pro interaktivní posuvníky.

Prvním krokem je definice základních proměnných, které získáme z PHP:

var minPrice = <?= $minPrice ?>;

var maxPrice = <?= $maxPrice ?>;

var minStart = <?= $filterMin ?>;

var maxStart = <?= $filterMax ?>;

Poté vytvoříme samotný posuvník:

document.addEventListener('DOMContentLoaded', function () {

var priceSlider = document.getElementById('priceSlider');

noUiSlider.create(priceSlider, {

start: [minStart, maxStart], // Nastavení počátečních hodnot

connect: true, // Propojení obou úchytů pro vizuální efekt

range: {

'min': minPrice,

'max': maxPrice

},

step: 1000, // Posun o 1000 Kč pro přehlednost

format: {

to: value => Math.round(value), // Zaokrouhlení hodnoty při posunu

from: value => Number(value) // Převod na číslo

}

});

// Synchronizace hodnot mezi posuvníkem a formulářem

var minInput = document.getElementById('minPriceInput');

var maxInput = document.getElementById('maxPriceInput');

var priceOutput = document.getElementById('priceOutput');

priceSlider.noUiSlider.on('update', function (values) {

minInput.value = values[0];

maxInput.value = values[1];

priceOutput.textContent = `${Number(values[0]).toLocaleString('cs-CZ')} Kč – ${Number(values[1]).toLocaleString('cs-CZ')} Kč`;

});

});

Teď si říkáte, co tento skript dělá? Hned vám to rozepíšu:

* Posuvník je inicializován při načtení stránky (DOMContentLoaded).
* Nastavení počátečních hodnot podle rozsahu cen z databáze (start).
* Automatické propojení hodnot (connect: true), což vizuálně spojí oba ovládací body.
* Zaokrouhlení hodnot (step: 1000), což zajistí, že uživatel nebude zadávat neplatné hodnoty.
* Synchronizace s formulářem, kdy jsou hodnoty posuvníku automaticky vloženy do <input> polí (minPriceInput, maxPriceInput).

Dále si řekneme něco o hover efektu při najetí na obrázky kol. Je to pěkný efekt, který dodává šarm webovému katalogu a vypadá to za mě hezky. Při najetí myší na obrázek kola se vizuálně zvětší, čímž se vytvoří atraktivní efekt zvýraznění. Tento efekt je jsem zrealizoval pomocí JavaScriptového event listeneru:

document.addEventListener('DOMContentLoaded', function () {

document.querySelectorAll('.card-img-top').forEach(function(img) {

img.addEventListener('mouseenter', function() {

img.style.transform = 'scale(1.08)';

img.style.transition = 'transform 0.3s ease-out';

img.style.zIndex = 10; // Předsazení obrázku pro lepší viditelnost

});

img.addEventListener('mouseleave', function() {

img.style.transform = 'scale(1)';

img.style.transition = 'transform 0.3s ease-in';

img.style.zIndex = 1;

});

});

});

* querySelectorAll('.card-img-top') vyhledá všechny obrázky kol.
* Při najetí myší (mouseenter) se obrázek zvětší (scale(1.08)), čímž vynikne.
* Při odjetí myší (mouseleave) se vrátí zpět na normální velikost (scale(1)).
* Změna vrstvení (zIndex = 10) zajistí, že obrázek nebude překryt ostatními prvky stránky.

Kromě cenového posuvníku by mělo být snadné aktualizovat hodnoty v ostatních filtrech bez nutnosti manuálního kliknutí. To umožňuje JavaScript:

document.addEventListener('DOMContentLoaded', function () {

var categorySelect = document.getElementById('category');

var brandSelect = document.getElementById('brand');

var sortSelect = document.getElementById('sort');

categorySelect.addEventListener('change', function () {

this.form.submit();

});

brandSelect.addEventListener('change', function () {

this.form.submit();

});

sortSelect.addEventListener('change', function () {

this.form.submit();

});

});

Při změně hodnoty v <select> filtrech, například kategorie nebo značky, se formulář automaticky odešle (this.form.submit()). To umožňuje okamžitou aktualizaci výsledků bez potřeby dalšího kliknutí na tlačítko „Filtrovat“. Stejný mechanismus se použije u řazení (sort), aby se seznam ihned přeuspořádal.

**HTML část:**

Struktura je postavena na Bootstrapu 5, což zajišťuje responzivní design a moderní vzhled. Kromě toho se využívají dynamické prvky jako cenový posuvník noUiSlider, filtrační formulář, výpis produktů a modální přihlašovací okno pro administrátory.

Hlavička stránky obsahuje základní meta informace a odkazy na externí knihovny:

<head>

<meta charset="UTF-8">

<meta name="viewport" content="width=device-width, initial-scale=1.0">

<title>Katalog Horských Kol</title>

<!-- Bootstrap 5 -->

<link href="https://cdn.jsdelivr.net/npm/bootstrap@5.3.3/dist/css/bootstrap.min.css" rel="stylesheet">

<script src="https://cdn.jsdelivr.net/npm/bootstrap@5.3.3/dist/js/bootstrap.bundle.min.js"></script>

<!-- Google Fonty -->

<link href="https://fonts.googleapis.com/css2?family=Inter:wght@400;600&display=swap" rel="stylesheet">

<!-- Vlastní CSS -->

<link rel="stylesheet" href="style.css">

<!-- noUiSlider (posuvník ceny) -->

<link rel="stylesheet" href="https://cdn.jsdelivr.net/npm/nouislider@15.7.0/dist/nouislider.min.css">

<script src="https://cdn.jsdelivr.net/npm/nouislider@15.7.0/dist/nouislider.min.js"></script>

</head>

Významy jednotlivých částí:

* Meta tagy nastavují správné kódování (UTF-8) a responzivitu (viewport).
* Bootstrap CSS & JS přidává předpřipravené styly a komponenty.
* Google Font "Inter" poskytuje čitelné a elegantní typografické provedení.
* noUiSlider umožňuje použití cenového posuvníku.

Tělo stránky obsahuje hlavní kontejner, který spravuje rozložení filtrů a katalogu kol.

<body>

<div class="container-fluid">

<h1 class="my-4">Katalog horských kol</h1>

<!-- Tlačítko pro mobilní zobrazení filtrů -->

<button class="btn btn-outline-dark w-100 mb-3 d-md-none" type="button" data-bs-toggle="collapse" data-bs-target="#filterSidebar">

Zobrazit / skrýt filtry

</button>

Opět vysvětlím, o co go. Takže container-fluid dělá široké rozvržení stránky na celou šířku.h1.my-4 je hlavní nadpis stránky s odstupem (margin-y 4). Bootstrap collapse tlačítko umožňuje skrývat filtry na mobilních zařízeních.

Filtrační formulář umožňuje uživatelům vyhledat kolo dle kategorie, značky, cenového rozpětí a řazení.

<div class="col-md-3">

<div class="collapse d-md-block show" id="filterSidebar">

<div class="filter-box">

<form method="GET" class="mb-3">

Zde je kód pro kategorie (select menu). Jsou zde dynamicky generované hodnoty kategorií přes PHP smyčku (foreach). Je zde taky ochrana XSS (htmlspecialchars).

<select name="category" id="category" class="form-select">

<option value="0">Vše</option>

<?php foreach ($categories as $category): ?>

<option value="<?= $category['id'] ?>" <?= $selectedCategory == $category['id'] ? 'selected' : '' ?>>

<?= htmlspecialchars($category['Category']) ?>

</option>

<?php endforeach; ?>

</select>

Cenový posuvník (noUiSlider) je použit pro intuitivní nastavení cenového rozpětí a taky protože se mi to prostě líbí a vidím to všude na různých mtb e-shopech.

<div id="priceSlider" class="mt-3 mb-3"></div>

<input type="hidden" id="minPriceInput" name="min\_price" value="<?= $filterMin ?>">

<input type="hidden" id="maxPriceInput" name="max\_price" value="<?= $filterMax ?>">

Dále zde máme výpis kol. Výsledky se zobrazují v Bootstrap kartách, které obsahují obrázek, název, cenu a tlačítko pro detail kola. Výpis je zde pomocí PHP cyklu ( foreach). Jsou zde použité Bootsrap karty, jak jsem již říkal pro vizuálně přehledné zobrazení. A je zde ochrana vyřešena pomocí (htmlspecialchars).

<?php foreach ($bikes as $bike): ?>

<div class="col-md-6 col-lg-4 mb-4">

<div class="card h-100">

<img src="img/<?= htmlspecialchars($bike['image\_main']) ?>" class="card-img-top" alt="Obrázek kola">

<div class="card-body">

<h5 class="card-title"><?= htmlspecialchars($bike['brand\_name']) ?> <?= htmlspecialchars($bike['model']) ?></h5>

<p class="card-text"><strong>Cena:</strong> <?= number\_format($bike['price'], 0, ',', ' ') ?> Kč</p>

<a href="detail.php?id=<?= $bike['id'] ?>" class="btn btn-primary">Zobrazit podrobnosti</a>

</div>

</div>

</div>

<?php endforeach; ?>

Přihlášení je pro administrátora udělané pomocí Bootstrap modalu. Je to pěkně přehledné a je zde použit formulář POST pro odeslání údajů na login.php.

<button class="btn btn-outline-primary" data-bs-toggle="modal" data-bs-target="#loginModal">

Admin / Login

</button>

<div class="modal fade" id="loginModal" tabindex="-1">

<div class="modal-dialog modal-sm modal-dialog-centered">

<div class="modal-content">

<form method="post" action="login.php">

<div class="modal-header">

<h5 class="modal-title">Přihlášení</h5>

### Spojení s databází

Soubor database.php slouží jako centrální modul pro navázání spojení s MySQL databází pomocí PDO. Ostatní části aplikace tento soubor volají (require 'database.php';) a následně pracují s instancí $pdo.

Zobrazení chyb (vývojové nastavení). ini\_set('display\_errors’ zapíná vypisování všech PHP chyb přímo na stránku.

// Zobrazení chyb při vývoji

ini\_set('display\_errors', 1);

ini\_set('display\_startup\_errors', 1);

error\_reporting(E\_ALL);

ini\_set('display\_startup\_errors', zajišťuje, že se zobrazí i chyby, které vznikají během startu PHP (např. při načítání rozšíření).

error\_reporting(E\_ALL) Nastavuje reportingová úroveň na všechny typy chyb, varování i upozornění. Tato nastavení je vhodné používat jen ve vývojovém prostředí, v produkci je vhodné chyby skrýt a logovat je do souboru.

// Připojení k databázi přes .ini soubor

$server = "websql.in.spsostrov.cz";

$data = parse\_ini\_file("../spso.mysql.ini");

Načtení souboru pro konfiguraci. $server Adresa databázového serveru (host).

parse\_ini\_file("../spso.mysql.ini") Čte konfigurační .ini soubor umístěný o úroveň výše mimo webroot. A zde je jak vypadá obsah spso.mysql.ini:

username = "db\_user"

password = "db\_pass"

database = "db\_name"

Výhodou .ini souboru je oddělení přihlašovacích údajů od zdrojového kódu a možnost spravovat je s vyšší úrovní přístupových práv. Dále zde máme uložení přihlašovacích údajů. $username – uživatelské jméno pro přihlášení k databázi. $password – je heslo k databázi, $database – je název databáze ke které se připojuji. Poslední část kódu v souboru database.php, vytvoření PDO připojení a Data Source Name (DNS).

try {

$pdo = new PDO(

"mysql:host=$server;dbname=$database;charset=utf8mb4",

$username,

$password,

[

PDO::ATTR\_ERRMODE => PDO::ERRMODE\_EXCEPTION,

PDO::ATTR\_DEFAULT\_FETCH\_MODE => PDO::FETCH\_ASSOC

]

);

} catch (PDOException $e) {

die("Připojení selhalo: " . $e->getMessage());

}

mysql:host=$server;dbname=$database;charset=utf8mb4. Host – adresa server.

### Design hlavní stránky

Soubor style.css definuje vizuální styl katalogu horských kol pomocí CSS custom properties, celoplošných pravidel, komponentních selektorů a responzivních úprav. Níže je rozdělení do sémantických bloků, přičemž každý blok popisuje konkrétní část kaskády a její účel. Začněme definicí barevných proměnných:

:root {

--dirt-bg: #23201c;

--dirt-bg-light: #3a3732;

--dirt-bg-dark: #181714;

--dirt-gray: #6e6a65;

--dirt-gray-light: #a9a29a;

--dirt-accent: #bfa97a;

--dirt-accent-dark: #9e8758;

--dirt-white: #eae6df;

}

V kořenovém selektoru: root jsou deklarovány CSS custom properties (proměnné), které definují základní paletu tmavého designu. Díky nim lze kontrastní barvy pozadí, textů a akcentů snadno měnit na jednom místě, aniž by bylo nutné upravovat více selektorů.

Dále zde mám globální styl těla stránky:

body {

background: var(--dirt-bg) url('img/neutral\_background.png') no-repeat center center fixed;

background-size: cover;

color: var(--dirt-white);

font-family: 'Inter', sans-serif;

}

Pravidlo pro body nastavuje tmavé pozadí (--dirt-bg) překryté pevným (fixed) obrázkem. Obrázek vyplní celý viewport (background-size: cover). Texty mají světle béžovou barvu (--dirt-white) a jako výchozí písmo se používá bezpatkové Inter, importované z Google Fonts.

.container-fluid {

max-width: 1440px;

margin: 0 auto;

padding: 2rem 1rem;

background: rgba(35, 32, 28, 0.96);

border-radius: 18px;

box-shadow: 0 8px 32px rgba(0,0,0,0.28);

}

.container-fluid omezuje největší šířku obsahu na 1440 px a centrovaně jej vkládá do viewportu. Vnitřní odsazení (2 rem nahoře/dole, 1 rem po stranách) vytváří prostor kolem obsahu. Poloprůhledné pozadí (RGBA) s jemným stínem a zakulacenými rohy odděluje obsah od pozadí stránky.

h1 {

font-weight: 800;

color: var(--dirt-accent);

text-shadow: 0 2px 4px rgba(0,0,0,0.25);

border-bottom: 2px solid var(--dirt-accent-dark);

padding-bottom: 1rem;

margin-bottom: 2rem;

}

Styling h1 dává nadpisu výrazný vzhled. Tučný řez (800), akcentní barva písma a jemný stín posilují čitelnost. Spodní ohraničení tmavším odstínem akcentu vizuálně odděluje hlavičku od zbytku stránky, přičemž padding a margin zaručují dostatečný prostor pod nadpisem.

.row {

margin-left: -2px;

margin-right: -2px;

}

.col-md-6, .col-lg-4 {

padding-left: 2px;

padding-right: 2px;

}

Tento blok upravuje výchozí mezery v Bootstrap gridu, aby mezi kartami bylo méně prázdného prostoru. Řádek (.row) má negativní margin, odpovídající drobnému vnitřnímu odsazení sloupců, které je nastaveno na 2 px.

.card {

background: var(--dirt-bg-light);

border: 1px solid var(--dirt-gray);

border-radius: 14px;

box-shadow: 0 6px 24px rgba(0,0,0,0.18);

display: flex;

flex-direction: column;

align-items: center;

min-height: 370px;

transition: box-shadow 0.3s, transform 0.3s;

overflow: hidden;

}

.card:hover {

transform: translateY(-5px);

box-shadow: 0 12px 32px rgba(191, 169, 122, 0.15);

border-color: var(--dirt-accent);

z-index: 2;

}

Kontejner. card používá světlejší tmavé pozadí a šedý okraj s mírným stínem. Flexbox zajišťuje vertikální rozložení obsahu a jednotnou minimální výšku. Přechodová animace umožňuje při hoveru pozvolné „zvednutí“ karty, zvětšení stínu a zvýraznění okraje akcentní barvou.

.card-img-top {

width: 98%;

height: 220px;

object-fit: contain;

margin: 1.1rem auto 0.5rem;

background: var(--dirt-bg-dark);

border-radius: 0.8rem;

border-bottom: 2px solid var(--dirt-gray);

transition: transform 0.3s cubic-bezier(.4,2,.3,1), border-color 0.3s;

will-change: transform;

}

.card-img-top:hover {

border-color: var(--dirt-accent);

}

Obrázek se zarovnává na střed a zachovává si poměr stran díky object-fit: contain. Tmavé podbarvení a zaoblené rohy ladí se zbytkem UI. Hover efekt zvýrazní spodní okraj obrázku akcentní barvou.

.card-title {

font-size: 1.13rem;

font-weight: 700;

color: var(--dirt-accent);

margin-bottom: 0.3rem;

}

.card-text {

font-size: 1rem;

margin-bottom: 0.7rem;

color: var(--dirt-gray-light);

}

.card-body {

flex: 1 1 auto;

padding: 0.3rem 0.5rem 1rem;

text-align: center;

}

Nadpis kola (.card-title) je tučný a v akcentní barvě, podtitulek s cenou (.card-text) je světle šedý. Kontejner .card-body centruje obsah a poskytuje rovnoměrný padding.

.btn-primary {

background: var(--dirt-accent);

border: none;

color: var(--dirt-bg-dark);

font-weight: 600;

letter-spacing: 0.02em;

border-radius: 0.7rem;

transition: background 0.15s, transform 0.2s;

}

.btn-primary:hover {

background: var(--dirt-accent-dark);

transform: scale(1.05);

}

.btn-outline-dark {

border-radius: .5rem;

border-color: var(--dirt-accent);

color: var(--dirt-accent);

background: none;

transition: background 0.2s, color 0.2s;

}

.btn-outline-dark:hover {

background: var(--dirt-accent);

color: var(--dirt-bg-dark);

}

Primární tlačítko (.btn-primary) používá akcentní barvu pozadí a tmavý text s mírným letter-spacingem. Hover stav mění odstín a mírně zvětší tlačítko. Obrysové tlačítko (.btn-outline-dark) má průhledné pozadí s barevným okrajem, který se po najetí vyplní.

.filter-box {

border-radius: 14px;

border: 1px solid var(--dirt-gray);

box-shadow: 0 4px 16px rgba(0,0,0,0.15);

background: var(--dirt-bg-light);

padding: 1.5rem 1rem 1rem;

margin-bottom: 2rem;

}

.form-label {

font-weight: 600;

margin-bottom: 0.5rem;

color: var(--dirt-accent);

}

Blok. filter-box odděluje filtrační panel vizuálně pomocí světlého pozadí, stínu a zaoblených rohů. .form-label zdůrazňuje popisky formuláře tučným řezem a akcentní barvou.

.form-select {

background: var(--dirt-bg-dark);

color: var(--dirt-white);

border: 1px solid var(--dirt-gray);

border-radius: 0.6rem;

font-size: 1rem;

transition: border-color 0.2s, box-shadow 0.2s;

}

.form-select:focus {

border-color: var(--dirt-accent);

box-shadow: 0 0 0 2px rgba(191, 169, 122, 0.25);

outline: none;

}

.form-select option {

background: var(--dirt-bg-light);

color: var(--dirt-white);

}

.form-select option:checked,

.form-select option:hover {

background: var(--dirt-accent-dark);

color: var(--dirt-bg-dark);

}

Výběrové menu je plně přizpůsobeno tmavému motivu: tmavé pozadí, světle béžový text a zvýraznění akcentní barvou při focusu či při přejíždění nabídkou.

input[type="range"] {

width: 100%;

background: transparent;

height: 8px;

margin: 0;

padding: 0;

accent-color: var(--dirt-accent);

}

/\* WebKit \*/

input[type="range"]::-webkit-slider-thumb { /\* … \*/ }

input[type="range"]::-webkit-slider-runnable-track { /\* … \*/ }

/\* Firefox \*/

input[type="range"]::-moz-range-thumb { /\* … \*/ }

input[type="range"]::-moz-range-track { /\* … \*/ }

/\* IE/Edge \*/

input[type="range"]::-ms-fill-lower,

input[type="range"]::-ms-fill-upper { /\* … \*/ }

input[type="range"]::-ms-thumb { /\* … \*/ }

/\* Focus & padding reset \*/

input[type="range"]:focus { outline: none; border: none; }

input[type="range"] { padding: 0; }

Standardní HTML slidery v různých prohlížečích dostávají jednotný vzhled: akcentní ovladač (thumb) se stínem, šedý track a odstraněné výchozí okraje či padding.

.range-wrapper {

display: flex;

gap: 0;

align-items: center;

margin-bottom: 0.5rem;

}

input[type=number]::-webkit-inner-spin-button,

input[type=number]::-webkit-outer-spin-button {

-webkit-appearance: none;

margin: 0;

}

.d-flex input[type=number] {

font-size: 1rem;

padding: 0.2rem 0.5rem;

border-radius: 0.5rem;

border: 1px solid var(--dirt-gray);

width: 100%;

background: var(--dirt-bg-dark);

color: var(--dirt-white);

}

.bg-light {

background-color: var(--dirt-bg-light) !important;

color: var(--dirt-white) !important;

}

.navbar-light .btn-outline-primary {

border-color: var(--dirt-accent);

color: var(--dirt-accent);

border-radius: .5rem;

transition: all 0.3s ease;

}

.navbar-light .btn-outline-primary:hover {

background-color: var(--dirt-accent);

color: var(--dirt-bg-dark);

}

.modal-content {

background-color: var(--dirt-bg-light);

color: var(--dirt-white);

border: 1px solid var(--dirt-accent-dark);

border-radius: 1rem;

}

Třída. range-wrapper slouží k horizontálnímu uspořádání slideru a číselných vstupů. Číselná pole ztratila spinnery a získala tmavé pozadí. Doplňkové utility přepisují některé Bootstrapové barvy na tmavé schéma, včetně modálního okna.

@media (max-width: 991px) {

.col-md-6, .col-lg-4 {

padding-left: 1px;

padding-right: 1px;

}

.container-fluid {

padding: 1rem 0.5rem;

}

.card-img-top {

height: 160px;

}

.card {

min-height: 300px;

}

}

@media (max-width: 768px) {

.filter-box {

margin-bottom: 1.2rem;

padding: 1.2rem 0.6rem;

}

}

Pro zobrazení na menších zařízeních jsou upraveny odsazení sloupců, velikost obrázků a minimální výška karet. Filtrační panel na úzkých obrazovkách získává upravený padding a margin, aby obsah zůstal přehledný i na mobilních telefonech.

#### 2.3.4 Soubor script.js

Soubor script.js obsahuje skript pro interaktivní práci s galerií obrázků, která využívá Bootstrap modal pro zvětšení obrázku. Tento skript inicializuje modalové okno, načítá miniatury obrázků, umožňuje jejich prohlížení a navigaci mezi obrázky pomocí tlačítek. Níže je rozdělen do jednotlivých funkčních bloků, přičemž každý blok obsahuje příslušný kód a jeho vysvětlení.

document.addEventListener('DOMContentLoaded', function () {

console.log('script.js načten');

Při načtení HTML dokumentu (DOMContentLoaded) se spustí skript, který zajišťuje správnou funkci galerie. Řádek console.log ('script.js načten'); slouží pro ladění – zobrazí zprávu v konzoli pro potvrzení, že se skript úspěšně načetl.

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

const modalImage = document.getElementById('modalImage');

const thumbnails = document.querySelectorAll('.gallery-thumb');

const prevBtn = document.getElementById('prevImage');

const nextBtn = document.getElementById('nextImage');

if (!modal || !modalImage || thumbnails.length === 0) {

console.error('Chybí prvky pro modal nebo miniatury.');

return;

}

Zde skript vyhledává v dokumentu:

* Modalové okno (imageModal) pro zobrazení obrázků.
* Obrázek uvnitř modalu (modalImage), který se dynamicky mění.
* Miniatury obrázků (gallery-thumb), které slouží jako seznam dostupných obrázků.
* Tlačítka pro navigaci (prevImage, nextImage), která umožňují procházení mezi obrázky.

Pokud některý z těchto prvků neexistuje (null nebo prázdné pole miniatur), skript vypíše chybu do konzole a ukončí svou činnost (return), což zabrání běhu s neúplnými daty.

let imageSources = Array.from(thumbnails).map(img => img.src);

let currentIndex = 0;

let bootstrapModal = new bootstrap.Modal(modal);

Miniatury se převedou na pole (Array.from(thumbnails)) a pomocí metody. map (img => img.src) se z nich vytvoří seznam cest (src) k obrázkům. Proměnná currentIndex uchovává index právě zobrazovaného obrázku. Bootstrap modal (bootstrapModal) je inicializován, aby bylo možné jej aktivovat skrze JavaScript.

thumbnails.forEach((thumb, index) => {

thumb.addEventListener('click', function () {

currentIndex = index;

modalImage.src = imageSources[currentIndex];

bootstrapModal.show();

});

});

Pomocí forEach se nastaví posluchač události (click) na každou miniaturu v galerii. Když uživatel klikne na obrázek tak se stane toto:

* Aktualizuje se index (currentIndex = index), aby odpovídal dané miniatuře.
* Změní se src obrázku v modalu, čímž se zobrazí požadovaný obrázek.
* Modal se otevře pomocí bootstrapModal.show(), což aktivuje efekt zvětšení.

Tento mechanismus umožňuje dynamické zobrazování obrázků bez nutnosti přesměrování na novou stránku.

prevBtn.addEventListener('click', function () {

currentIndex = (currentIndex - 1 + imageSources.length) % imageSources.length;

modalImage.src = imageSources[currentIndex];

});

Tlačítko „Předchozí obrázek“ (prevBtn) reaguje na kliknutí.Sníží se index obrázku (currentIndex - 1). Použije se operace % imageSources.length, která zajistí, že při dosažení prvního obrázku se přechod přesměruje na poslední. Změní se src obrázku v modalu, což zobrazí předchozí obrázek v seznamu.

nextBtn.addEventListener('click', function () {

currentIndex = (currentIndex + 1) % imageSources.length;

modalImage.src = imageSources[currentIndex];

});

Tlačítko „Další obrázek“ (nextBtn) funguje podobně. Zvýší se index obrázku (currentIndex + 1). Pokud se dostane za poslední obrázek, pomocí operace % imageSources.length se nastaví zpět na první. Změní se src obrázku v modalu, čímž se posune na další obrázek.

Závěrem. Soubor script.js efektivně řídí zobrazení galerie obrázků pomocí modalového okna Bootstrapu. Jeho funkcionalita zahrnuje: Načtení seznamu obrázků z miniatur Interaktivní kliknutí na miniaturu pro zvětšení obrázku Navigaci mezi obrázky pomocí tlačítek Zpracování chyb při neexistenci prvků. Tento skript zajišťuje elegantní procházení obrázků v galerii bez zbytečného obnovování stránky, čímž uživateli poskytuje plynulý zážitek.

#### Login a logout administrátora

Soubor login.php slouží k ověření administrátorského přihlášení na webovou aplikaci. Používá session k uchování stavu přihlášení, PDO pro bezpečnou komunikaci s databází a HTTP přesměrování pro správnou navigaci uživatele podle výsledku ověření.

session\_start();

require 'database.php';

V první části kódu se inicializuje session (session\_start();), což umožňuje uchovávat stav přihlášení mezi stránkami. Poté se importuje soubor database.php, který obsahuje připojení k databázi pomocí PDO.

Tímto se zajistí, že můžeme bezpečně komunikovat s databází bez nutnosti opakovaně definovat připojení.

if ($\_SERVER['REQUEST\_METHOD'] === 'POST') {

$username = $\_POST['username'] ?? '';

$password = $\_POST['password'] ?? '';

Ověřujeme, zda požadavek na přihlášení byl odeslán metodou POST. Pokud ano, získáme username a password, které uživatel zadal do formuláře.

Pokud nejsou definovány ($\_POST['username'] ?? ''), nastaví se na prázdný řetězec, což zabrání chybám při práci s těmito hodnotami.

$stmt = $pdo->prepare("SELECT \* FROM admins WHERE username = ?");

$stmt->execute([$username]);

$admin = $stmt->fetch();

Používáme připravený SQL dotaz (prepare()), abychom zabránili SQL injection. Tento dotaz vybírá všechny sloupce (SELECT \*) ze tabulky admins, kde se username rovná hodnotě z formuláře. Poté execute([$username]) spouští dotaz s bezpečně vloženým parametrem. Výsledek se získá pomocí fetch(), čímž se data administrátora uloží do proměnné $admin.

if ($admin && $password === $admin['password']) {

$\_SESSION['admin\_logged\_in'] = true;

$\_SESSION['admin\_username'] = $username;

header("Location: admin\_dashboard.php");

exit;

}

Pokud existuje záznam v databázi ($admin) a heslo odpovídá ($password === $admin['password']), nastavíme session proměnné:

* $\_SESSION['admin\_logged\_in'] = true; → označuje administrátora jako přihlášeného.
* $\_SESSION['admin\_username'] = $username; → uchovává jméno přihlášeného administrátora.

Po správném ověření se uživatel přesměruje na administrátorskou stránku (admin\_dashboard.php), kde může spravovat katalog kol.

else {

header("Location: access.php?login=fail");

exit;

}

Pokud přihlášení selže (uživatel neexistuje nebo zadal nesprávné heslo), uživatel se přesměruje zpět na access.php s parametrem login=fail, který může být využit pro zobrazení chybové zprávy.

Soubor login.php umožňuje přihlášení administrátora pomocí: Session pro uchování přihlášení PDO pro bezpečnou práci s databází Přesměrování uživatele podle výsledku přihlášení. Nyní se přesuňme k souboru logout.php.

Soubor logout.php slouží k odhlášení administrátora z webové aplikace. Jeho účel je ukončit aktivní relaci (session), odstranit uložené údaje o přihlášení a přesměrovat uživatele zpět na hlavní stránku katalogu (access.php). Níže jsou jednotlivé kroky podrobněji popsány.

session\_start();

Tento příkaz zajistí, že se aktivuje aktuální session, aby ji bylo možné následně ukončit. Bez tohoto kroku by nebylo možné upravovat session data a správně odstranit přihlašovací stav uživatele.

session\_destroy();

Po zavolání session\_destroy(); se trvale odstraní všechna data session. To znamená:

* Uživatel už nebude přihlášen ($\_SESSION['admin\_logged\_in'] přestane existovat).
* Session proměnné jako $\_SESSION['admin\_username'] se vymažou.
* Všechny hodnoty spojené s administrátorem budou zapomenuty.

Jediné, co zůstane aktivní, je session ID, které se obvykle znovu inicializuje při další návštěvě stránky.

header('Location: access.php');

exit;

Po zničení session je uživatel automaticky přesměrován na access.php, což je hlavní vstupní stránka katalogu. To zajistí, že uživatel nebude mít přístup k administrátorským funkcím bez opětovného přihlášení. Příkaz exit; zabrání jakémukoli dalšímu vykonávání kódu po přesměrování. Závěrem, soubor logout.php je velmi jednoduchý, ale klíčový pro správu bezpečného odhlášení administrátorů. Jeho funkcionalita zahrnuje: Zahájení session – nutné pro její správné odstranění. Zničení session – vymazání uložených dat. Přesměrování na hlavní stránku – aby administrátor již neměl přístup k chráněným sekcím.

## Detail horského kola

Tento skript v jazyce PHP slouží k dynamickému načtení detailu vybraného jízdního kola z databáze a následnému zobrazení těchto informací na stránce. Celá aplikace kombinuje serverovou logiku (PHP) s klientským zobrazením (HTML, CSS, JavaScript) a využívá framework Bootstrap pro responzivní design a různé efekty.

### Načtení dat z databáze

<?php

require 'database.php';

$bikeId = isset($\_GET['id']) ? (int) $\_GET['id'] : 0;

$stmt = $pdo->prepare("SELECT bikes.\*, brands.Brand AS brand\_name, categories.Category AS category\_name

FROM bikes

JOIN brands ON bikes.brand\_id = brands.id

JOIN categories ON bikes.category\_id = categories.id

WHERE bikes.id = :id");

$stmt->execute(['id' => $bikeId]);

$bike = $stmt->fetch(PDO::FETCH\_ASSOC);

if (!$bike) {

echo "Kolo nebylo nalezeno.";

exit;

}

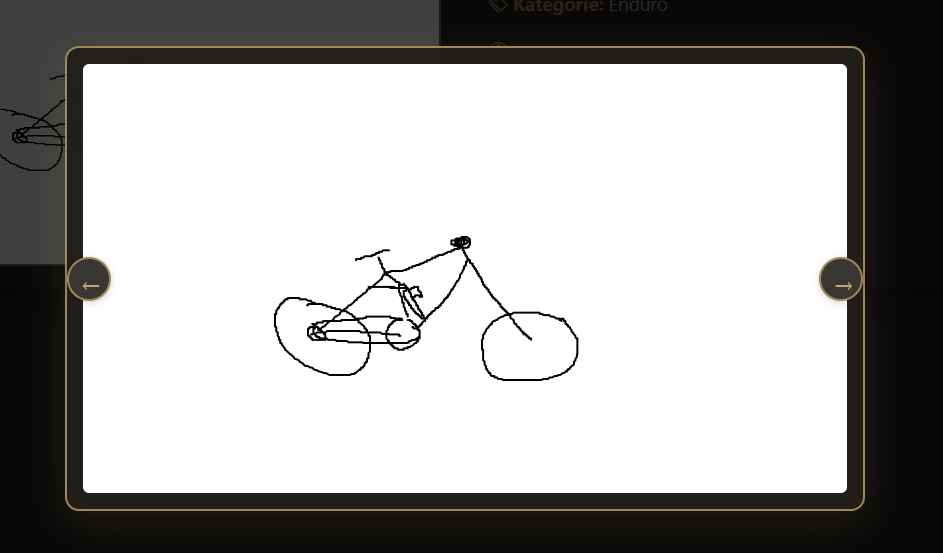
$imageGallery = explode(',', $bike['image\_gallery']);

?>

Popis toho, co tento PHP scrit dělá: Začneme s načtením souboru, který obsahuje konfiguraci a inicializaci připojení k databázi pomocí PDO. Hodnota parametru id z url ($\_GET) je převedena na celé číslo a uložená do proměnné $bikeId. Pokud parametr není poskytnut, nastaví se výchozí hodnota 0. Dále se připraví SQL dotaz, který načítá údaje o kole z tabulky bikes a zároveň pomocí JOIN načítá odpovídající značku z tabulky brands a kategorii z tabulky categories. Parametr: id je vázán na hodnotu $bikeId. Pokud dotaz nevrátí žádná data (kolo s daným ID neexistuje), script vypíše zprávu „Kolo nebylo nalezeno“. A ukončí běh. Hodnota ve sloupci image\_gallery, která obsahuje seznam všech názvů obrázků, oddělených čárkou, se pomocí funkce explode převede na pole, které se dá dále využít při vykreslování galerie.

### Vzhled a rozvržení

A teď aby to nebylo jenom kódově tak si podle **Obrázek 3 Detail vybraného kola** probereme, co a jak. V levé části stránky je implementovaná galerie s využitím karuselu, o kterém jsme se učili ve škole. Obrázky jsou načítány dynamicky z pole, které bylo vytvořeno z databáze. Díky zabudované podpoře Bootstrapu jsou obrázky prezentovány přehledně, s možností jejich interaktivního zvětšení v modálním okně.



Obrázek 7 Modální okno detailu kola

V pravé části stránky se zobrazují klíčové údaje o kole, značka, model, kategorie, barva cena a detailní popis. Data jsou bezpečně zpracovávana, aby nedošlo k nějakému špatnému zobrazení. Vše je formátované pro přehlednost a je doplněno o Bootstrap icons, pro lepší přehlednost. Pro zvýšení uživatelské přívětivosti je přidána interaktivita do galerie. Při kliknutí na libovolný obrázek v karuselu se otevře modální okno, kde se obrázek zobrazí ve zvětšeném formátu. Uživatel tak může využít navigační tlačítka pro procházení obrázků vpřed i vzad, což se děje pomocí JavaScriptu, který dynamicky mění obsah modálního okna. Cyklus, kdy se obrázky přehrávají, je implementován tak, že při dosažení posledního obrázku se automaticky přejde z posledního znova na první.

<!-- Modal -->

<div class="modal fade" id="imageModal" tabindex="-1" aria-hidden="true">

<div class="modal-dialog modal-dialog-centered modal-lg">

<div class="modal-content">

<div class="modal-body text-center position-relative">

<button id="prevImage" class="btn btn-nav position-absolute top-50 start-0 translate-middle-y">&larr;</button>

<img id="modalImage" src="" class="img-fluid rounded" alt="Zvětšený obrázek">

<button id="nextImage" class="btn btn-nav position-absolute top-50 end-0 translate-middle-y">&rarr;</button>

</div>

</div>

</div>

</div>

<script src="https://cdn.jsdelivr.net/npm/bootstrap@5.3.3/dist/js/bootstrap.bundle.min.js"></script>

<script>

const gallery = <?= json\_encode(array\_map('htmlspecialchars', $imageGallery)) ?>;

let currentIndex = 0;

const modalImage = document.getElementById('modalImage');

const prevBtn = document.getElementById('prevImage');

const nextBtn = document.getElementById('nextImage');

// Při kliknutí na obrázek v karuselu se zobrazí modal s daným obrázkem

document.querySelectorAll('.carousel-item img').forEach((img, index) => {

img.addEventListener('click', () => {

currentIndex = index;

modalImage.src = img.src;

const modal = new bootstrap.Modal(document.getElementById('imageModal'));

modal.show();

});

});

// Předchozí obrázek

prevBtn.addEventListener('click', () => {

currentIndex = (currentIndex - 1 + gallery.length) % gallery.length;

modalImage.src = 'img/' + gallery[currentIndex];

});

// Další obrázek

nextBtn.addEventListener('click', () => {

currentIndex = (currentIndex + 1) % gallery.length;

modalImage.src = 'img/' + gallery[currentIndex];

});

</script>

Modal (vyskakovací okno) je definován pomocí Bootstrap komponenty. Obsahuje tlačítka pro navigaci vlevo a vpravo a <img> element, který slouží k zobrazení zvětšeného obrázku z galerie. Pomocí Javascriptu je inicializováná galerie: Pomocí json\_encode se serverově vytvořená pole obrázků převedou do JavaScript proměnné gallery.

Event listener na obrázky: Každý obrázek v karuselu má přiřazený listener. Po kliknutí se nastaví aktuální index a změní se zdroj obrázku v modalu. Následně se modal otevře.

Jak mám vyřešené avigační funkce:

Při kliknutí na tlačítko pro předchozí obrázek se index sníží o 1 a vypočítá se pomocí modulo operace pro zajištění cyklického přechodu.

Při kliknutí na tlačítko pro další obrázek se index zvýší o 1, opět pomocí modulo operace zajišťuje cyklickou změnu.

Celý systém je navržen s ohledem na oddělení databázové logiky a prezentační vrstvy. Použití PHP pro serverovou stránku a Bootstrapu pro klientské zobrazení přispívá k čisté architektuře a snadné údržbě. Celkové řešení zajišťuje, že uživatelé obdrží přesné a na míru šité informace o kole, a zároveň mají možnost interaktivně prohlížet detailní fotografie bez nutnosti opuštění stránky.

## Administrátor

V této části se budeme věnovat administrátorské části katalogu. Řekneme si, jak se administrátor přihlašuje do své admin sekce, a co zde může dělat. Na obrázku: **Obrázek 8 přihlášení pro administrátora** vidíte přihlašovací formulář pro administrátora. Uživatelské jméno je: (admin) a heslo: (admin123).



Obrázek 8 přihlášení pro administrátora

### Dashboard

#### Zkouška 1

Graf 1 Stanovisko od roku 1890

Obrázek 9 HDD 3,5 palců 5400 otáček

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |

Tabulka 1 Tabulka srovnání disků

## Závěr

# Citovaná literatura

1. **PŠENŠÍKOVÁ, Jana.** *Algoritmizace.* Kralice na Hané : Comuter Press, 2009. ISBN-90-80-70-20-10.

2. **ŠKODA, Jiří a DOULÍK, Pavel.** *Psychodidaktika : metody efektivního a smysluplného učení a vyučování.* Pedagogika. Praha : Grada, 2011. str. 206. 978-80-247-3341-8.

3. **ZUNA, Pavel.** Den, kdy byla zveřejněna první webová stránka (6. srpen). *stream.* [Online] 06. 08 2015. [Citace: 12. 01 2016.] https://www.stream.cz/slavnedny/10006705-den-kdy-byla-zverejnena-prvni-webova-stranka-6-srpen.

4. O škole Charakteristika školy. *Střední průmyslová škola Ostrov.* [Online] [Citace: 16. 3 2016.] http://spsostrov.cz/charakteristika-skoly-1.

5. **BÍLEK, Miroslav.** Tvorba webových stránek - Studijní text. *SPŠ Mohelnice.* [Online] [Citace: 29. 05 2015.] http://www.spsemoh.cz/vyuka/sict/09/s09t.ppt.

6. JavaScript Tutorial. *w3schools.com.* [Online] [Citace: 26. 02 2016.] http://www.w3schools.com/js/default.asp.

7. HyperText Markup Language. *Wikipedie.* [Online] [Citace: 25. 04 2016.] https://cs.wikipedia.org/wiki/HyperText\_Markup\_Language.

# Seznam obrázků

[Obrázek 1Vzhled Filtrů a třídění 3](#_Toc199840423)

[Obrázek 2Bootsrap Card na webu 3](#_Toc199840424)

[Obrázek 3 Detail vybraného kola 4](#_Toc199840425)

[Obrázek 4 Tabulka bikes v admineru 3](#_Toc199840426)

[Obrázek 5 tabulka categories v admineru 4](#_Toc199840427)

[Obrázek 6 tabulka brands v admineru 5](#_Toc199840428)

[Obrázek 7 Modální okno detailu kola 32](#_Toc199840429)

[Obrázek 8 přihlášení pro administrátora 35](#_Toc199840430)

[Obrázek 9 HDD 3,5 palců 5400 otáček 36](#_Toc199840431)

# Seznam tabulek

[Tabulka 1 Tabulka srovnání disků 36](#_Toc199832803)

# Seznam grafů

[Graf 1Stanovisko od roku 1890 31](#_Toc481138632)

# Seznam příloh

Obrázek 1Vzhled Filtrů a třídění……………………………………………………………3

Obrázek 2Bootsrap Card na webu…………………………………………………….…….3

[Obrázek 3Detail vybraného kola 4](#_Toc199832624)

# Seznam použitého softwaru

# Seznam použitých odborných výrazů