



STŘEDNÍ PRŮMYSLOVÁ ŠKOLA
OBCHODNÍ AKADEMIE
JAZYKOVÁ ŠKOLA
FRÝDEK-MÍSTEK

Příspěvková organizace
Moravskoslezského kraje



MATURITNÍ ZKOUŠKA

PRAKTICKÁ ZKOUŠKA Z ODBORNÝCH PŘEDMĚTŮ

Téma č.3

Fotografický web a kompletní administrační systém

Obor vzdělání: **18 – 20 – M/01 Informační technologie**

Třída: **4. IT** Autor práce: **Jiří Vala**

Školní rok: **2021/22** Vedoucí učitel práce: **Ing. Jiří Sumbal**

Prohlášení Autora

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

Ve Frýdku-Místku, dne: podpis

Zadání práce

1. Návrh a koncepce webu - shrnutí požadavků zákazníka a návrh řešení
2. Použité platformy, technologie a software
3. Vývoj webové stránky
4. Vývoj administračního systému
5. Hosting
6. Vytvoření vlastního webového serveru a virtuálního webu na bázi virtuálního počítače

Anotace

Tato práce se zabývá průběhem vývoje webové stránky pro amatérského fotografa ve formě virtuální galerie a jejího administračního systému. Popisuje použité technologie a postupy ve vývoji spolu s konfigurací jednotlivých služeb klíčových ke spuštění celého projektu na produkční server a jeho správné a dlouhodobě nezávadné fungování. Cílem práce bylo vytvořit funkční produkt v podobě webu a jeho administračním systémem se všemi aspekty komerčního projektu pro koncového zákazníka, který jej bude používat jako své virtuální portfolio.

Anotation

This thesis focuses on the development process of a website for an amateur photographer in a form of virtual gallery and it's content management system. Describes used technologies and procedures used in the development as well as configuration of individual services mandatory for deployment of the whole project on a production server and it's seamless and longterm functioning.

Klíčová slova

webová stránka; webová aplikace; administrační systém; webová prezentace; webová galerie; vývoj; javascript; react; linux; webhosting

Obsah

1	Vymezení pojmu	6
1.1	Seznam použitých zkratek	8
2	Úvod	9
3	Klíčové vlastnosti a požadavky projektu	10
3.1	Serverless architektura	11
3.2	Moderní technologie	12
3.2.1	Frontend a UI/UX	12
3.3	Backend	14
3.4	Hosting	16
4	Struktura projektu	17
5	Průběh vývoje	18
5.1	Komunikace se zákazníkem	19
5.2	Organizace úkolů a změn k implementaci	19
6	Webová prezentace	20
7	Administrační systém	23
8	VPS - Virtual Private Server	26
8.1	VPS provideři	26
8.2	Konfigurace VPS	27
8.2.1	Operační systém GNU/Linux	27
8.2.2	Vytvoření developer uživatele	28
8.2.3	Konfigurace prostředí	29
8.3	Konfigurace CI/CD	31

8.3.1	Naklonování projektu na lokální úložiště	32
8.3.2	Nginx konfigurace pro web i administrační systém	33
9	Závěr	37
10	Seznam použitého softwaru	38
11	Seznam obrázků	39
12	Citace	40

1 Vymezení pojmu

1. **Content Management System** - Administrační systém, je systém, který umožňuje manipulaci s daty a obsahem webové stránky či webové (ale i desktopové) aplikace.
2. **Hosting (Webhosting)** - Znamená pronájem prostoru na cizím webovém serveru za účelem spuštění webové stránky či aplikace.
3. **Continuous Deployment (CI/CD)** - CI - Continuous Intergration a CD - Continuous Delivery je proces, který umožňuje nepřetržité dodávání aktuální verze projektu k testování či produkčnímu spuštění. V praxi to znamená, že pokud tým vývojářů provede změnu a uloží ji do remote repositáře (místo pro ukládání zdrojových kódů v cloudu), tato změna se automaticky projeví (deployne) na produkční server (server, na kterém je spuštěna aktuální verze projektu a je poskytována uživatelům z venčí).
4. **Fetching** - Proces během kterého se "stahuje" data z databáze. Tento anglický výraz doslova znamená "přinést".
5. **Backend(Server)** - Logická část aplikace nebo webu. Požadavky, které uživatel aplikace provede (například přejmenování záznamu v databázi) a data (například všechny záznamy), které si vyžádá, tento server nejdříve zpracuje, případně vytáhne z databáze a pak je odešle uživateli aplikace, která je následně zobrazí, či případně upraví v databázi.
6. **Frontend** - Jedná se o grafické uživatelské rozhraní, v rámci webových aplikací a stránek většinou nejčastěji pomocí značkovacího jazyka HTML, stylovacího jazyka CSS a scriptovacího jazyka Javascript.
7. **Cloud Computing Provider** - Poskytovatel cloudových služeb a infrastruktury (serverů).

8. **Public/Private API** - API je soubor procedur, funkcí a protokolů zajišťující komunikaci mezi dvěma platformami, které si vzájemně vyměňují data (například dvě rozdílné aplikace). K public API má přístup kdokoliv, k private API jen autentifikovaní uživatele například pomocí unikátního ID tokenu.
9. **DDOS útok** - Distributed Denial Of Service - je útok jehož cílem je pomocí sítě mnoha virtuálních počítačů, které v jednu chvíli budou provádět požadavky na daný server, vyřadit tento server z provozu, jelikož nezvládne nápor takového počtu požadavků a zhroutí se.
10. **Framework** - Jedná se o podpůrnou softwarovou strukturu, která zjednoduší vývoj a organizaci projektu v daném jazyce. Může obsahovat další funkce, podpůrné programy nebo vzory a doporučené postupy ve vývoji.
11. **Knihovna** - Jedná se souhrn procedur a funkcí, často avšak také konstant a datových typů, které usnadňují vývojáři práci tím, že může použít již hotový kód.
12. **Open-source** - Jedná se o projekt s otevřeným zdrojovým kódem, což znamená, že na jeho vývoji se může podílet každý a každý si jej může stáhnout. Popularita open-source projektů v posledních letech velmi roste.
13. **CRUD operace** - Create, read, update, delete (vytvořit, číst, změnit, smazat) jsou čtyři základní operace se záznamem v databázové tabulce.
14. **Shell** - Jedná se o vrstvu operačního systému, která provádí příkazy a spouští programy.
15. **Pagination** - Jedná se o proces rozdělování obsahu na jednotlivé stránky a podstránky. Nejčastěji ji vidíme ve formě čísel na konci webové stránky.
16. **Dropdown menu** - Jedná se o list záznamů, který se zobrazí až ve chvíli, kdy uživatel nějakým způsobem provede interakci s tímto menu, například kliknutím myši.
17. **Algoritmus** - Proces jednotlivých příkazů za účelem splnění zadaného úkolu.
18. **Push notifikace** - Notifikace, které se zobrazují uživateli na mobilním zařízení v notifikační listě nebo na zamýkací obrazovce.

1.1 Seznam použitých zkratek

1. **DMP** - Dlouhodobá maturitní práce
2. **CMS** - Content Management System
3. **VPS** - Virtual Private Server
4. **CI** - Continuous Integratiaon
5. **CD** - Continuous Deployment
6. **API** - Application Programming Interface
7. **GNU** - GNU's Not Unix
8. **GUI** - Graphical User Interface
9. **CLI** - Command Line Interface
10. **UI/UX** - User Interface/User Experience

2 Úvod

Pan Milan Bureš st. (dále jen fotograf nebo zákazník) je amatérský fotograf, který si již před pár lety nechal na zakázku vytvořit webovou stránku, na které by prezentoval své fotografie, zachycující náhodné okamžiky jeho života. Nicméně po stupem času se jeho web i administrační systém stali zastaralými a bylo potřeba celý systém zmodernizovat i vzhledem k faktu, že jeho webová stránka fungovala pomocí Adobe Flash Player, kterému skončila podpora a není již podporovaný moderními prohlížeči. Proto mě v rámci dlouhodobé maturitní práce zprostředkované mým garantem kontaktoval a požádal, zda-li bych jeho webovou prezentaci nemohl modernizovat. Mým úkolem bylo vytvořit web, který bude sloužit jako virtuální galerie jeho fotografií a bude zobrazitelná a funkční na všech zařízeních, a administrační systém (dále jen CMS), který bude umožňovat nahrávání, správu a manipulaci fotografií či alb, ve kterých se jednotlivé fotografie seskupují. Pro potřeby mé maturitní práce bylo potřeba, aby projekt obsahoval i část jiného předmětu, než jen programování (celý systém jsem naprogramoval) a databáze (data a fotografie, které fotograf nahraje/uloží). Rozhodl jsem se tedy, že si vytvořím vlastní server běžící na operačním systému Linux, na kterém budu hostovat vlastní instanci celého systému a bude obsahovat funkctionalitu tzv. Continuous Deploymentu, což přesně kopíruje funkcionality komerční instance projektu určené jen pro účely zákazníka.

3 Klíčové vlastnosti a požadavky projektu

Během vývoje webové prezentace byl velký důraz kladen na to, aby se co nejvíce podobala stávající verzi. Ne veškeré elementy webu byly replikovatelné a také jsem se chtěl co nejvíce řídit pravidly dobrého webu. Pro porovnání, zde jsou screenshoty jednotlivých webových stránek.

ROZEPSAT



Úvodem
Nejnovější
Fotografie do 1989
Fotografie po 1989
Soubory
Biografie
Výstavy
Kontakt
English



2013 © Copyright Milan Bureš st.
Webprogramming by All Developers Studio
Administrace

Původní webová stránka



Úvodem
Nejnovější
Soubory
Výstavy
Biografie
Kontakt



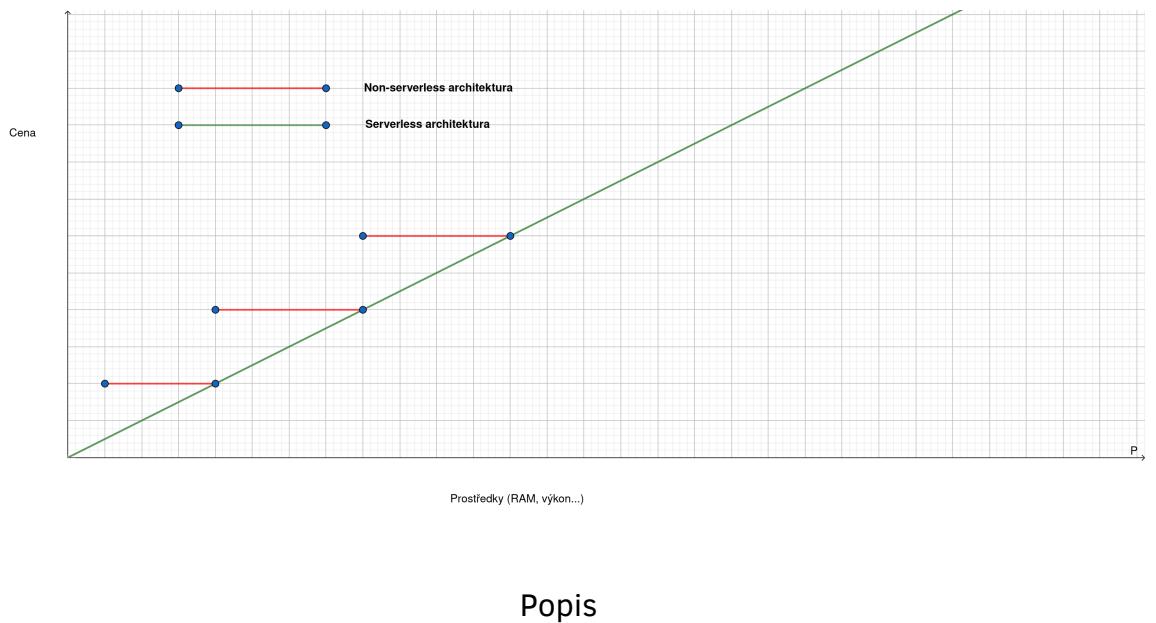
2021 © Mgr. Milan Bureš st.
Made with ❤ by Orexin.

Mnou vytvořená webová stránka

3.1 Serverless architektura

Ačkoliv je samozřejmostí, že je jak webová prezentace tak CMS hostovaná na nějakém fyzickém serveru, k požadavkům jako fetching dat z databáze nepoužívám žádný další (backend) server. To znamená, že jsem sice odkázaný kompletně na svého cloud computing providera, což je v mé případě Firebase od Google, ale na druhou stranu se nemusím (a ani zákazník) starat o údržbu vlastního serveru a ani jeho první konfiguraci, vše je tzv. out-of-the-box připravené k použití. Díky serverless architektuře jsem nemusel vytvářet žádný backend server s private API, pomocí kterého bych dostával data z databáze a cloud storage do webové prezentace nebo administračního systému. Serverless architektura má mnoho výhod, jedna z těch hlavních a největších je samozřejmě absence potřeby konfigurovat a později spravovat vlastní server, jelikož to kompletně zprostředkovává daný cloud computing provider, ale také celková cena "pronájmu", jelikož ta se počítá podle počtu požadavků na danou službu (například fetchování dat z databáze), bezpečnost, jelikož jsou tyto architektury zabezpečené například proti DDOS útokům, ale také celková

škálovatelnost celého systému. Cloud computing provider automaticky škáluje prostředky (úložistě, výpočetní výkon apod.), které jsou k dispozici pro daný systém.



Na grafu jde vidět, že pokud si zvolíme cestu non-serverless architektury, musíme s narůstajícími požadavky výkon zvyšovat, což ale v praxi znamená dočasné odstavení systému od provozu a dodatečnou správu hardwaru a tento výpočetní výkon se nemění, pokud se opět nevylepší, zatímco serverless řešení se automaticky škáluje.

3.2 Moderní technologie

Jelikož byla původní webová stránka i administrační systém postavena na poměrně zastaralých technologiích, bylo potřeba modernizovat a během vývoje jsem používal jen ty nejmodernější a mezi vývojáři velmi rozšířené technologie.

3.2.1 Frontend a UI/UX

Jedná se o grafické uživatelské rozhraní, v rámci webových aplikací většinou nejčastěji pomocí značkovacího jazyka HTML, stylovacího jazyka CSS a scriptovacího jazyka Javascript. Nicméně v dnešní době na tyto technologie, které existují delší dobu, existuje mnoho frameworků a knihoven, které nejen zjednodušují práci vývojářům, ale také celkově zlepšují performance aplikace či webu, která je díky

tomu rychlejší, intuitivnější a použitelnější, což ve výsledku znamená, že například návštěvník webové stránky neztratí zájem, jelikož se stránka nebude dlouze načítat. UI (user interface - uživatelské prostředí) a UX (user experience - v hrubém překladu uživatelská zkušenost) je sada pravidel a procedur, kterém mají za úkol zjednodušit uživatelskou interakci se systémem tak, aby se uživateli produkt používal co nejsnáze a byl co nejvíce intuitivní. Během vývoje administračního systému jsem se inspiroval návrhem, který jsem našel na designovém webu Dribble.com.

3.2.1.1 HTML,CSS a Javascript

Jedná se o základní technologie, se kterými se dnes vyvíjejí webové stránky či aplikace a všechny ostatní technologie si z nich něco vzaly. Základním stavebním kamenem webové stránky i aplikace je HTML dokument, který obsahuje všechny elementy, které má web zobrazovat. Tyto elementy nastyluje stylovací jazyk CSS, což znamená, že v CSS dokumentu vybereme jednotlivý HTML element pomocí CSS selektoru a nastavíme mu například jinou barvu, šířku, či například nastavíme animaci, která se spustí po tom, co uživatel přes tento element přejede myší. Pokud má mít web nějakou funkcionality, například po zadání dvou čísel vypočítat součet, použijeme Javascript. Ten je vlastně logickým mozkem celého webu a stará se o tyto početní operace, získávání dat z databáze a jejich následné vložení do struktury HTML dokumentu (DOM) nebo interaktivitu s uživatelem.

3.2.1.2 React.js

React.js je Javascriptový framework vyvíjený firmou Meta (Facebook) a open-source komunitou vývojářů po celém světě a k dnešnímu dni se jedná o nejvíce používaný framework na světě pro jeho jednoduchost a modularitu. Je optimální pro tvorby SPA (single page aplikací), jako je například administrační systém, protože velmi dobře pracuje s rychle měnícími se daty. Během vývoje react aplikace se celá aplikace rozdělí na jednotlivé komponenty, což má za následek jednoduchý a přehledný zdrojový kód a celou strukturu projektu, což je potřebné u velkého a komplexního projektu a já sám jsem během vývoje pocítil výrazný rozdíl v jednoduchosti vývoje oproti samotnému vanilla Javascriptu (bez použití frameworku či

knihovny.)

[react codeblock]

3.2.1.3 SASS/SCSS

SASS je preprocessor stylovacího jazyka CSS, což znamená, že styly psané v SCSS (sassy css) zpracuje a překompiluje to klasického CSS. SCSS je další z mnoha stylovacích jazyků a jedná se o přímé rozšíření klasického CSS, přidává například proměnné, nesting, což znamená, že jednotlivé selektory můžeme psát přímo hierarchicky do sebe a pod sebe, což zlepšuje přehlednost jednotlivých stylů a například tzv. mixins, což jsou vlastně bloky stylů, které se dají použít vícekrát, což snižuje počet stylů, které musí vývojář napsat.

[css codeblock]

[scss codeblock]

3.3 Backend

3.3.0.1 Node.js

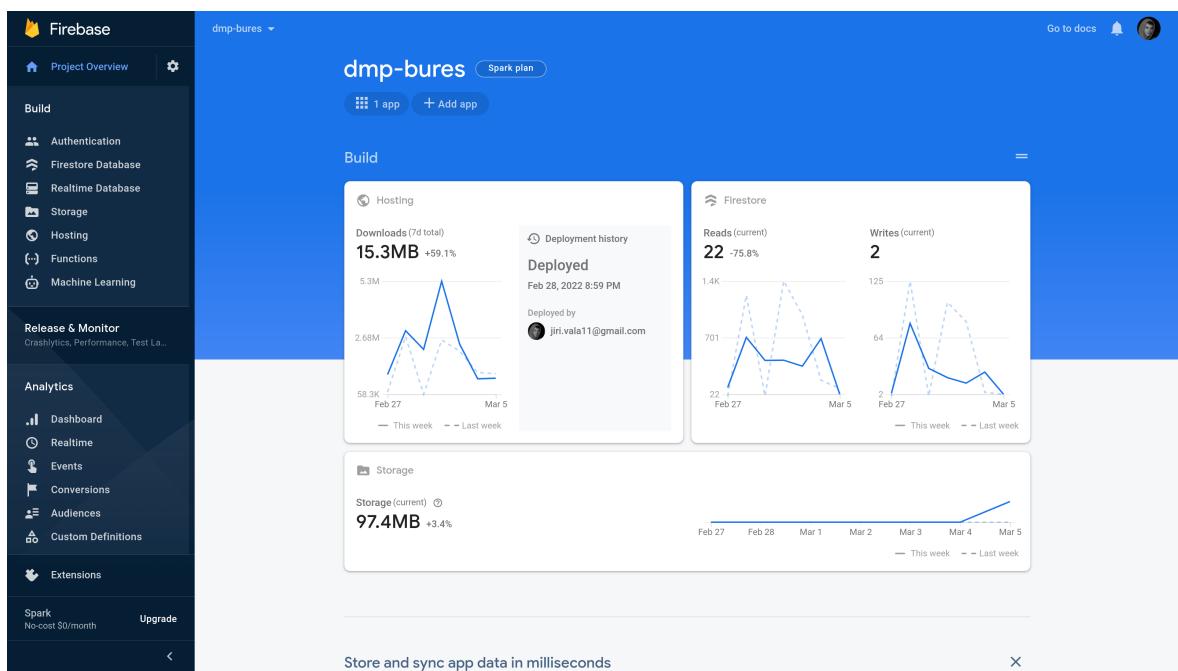
Node.js je tzv. runtime pro Javascript určený k vytváření vysoce škálovatelných backend aplikací a webových serverů. Toto má společné např. s jazykem PHP, který má stejné zaměření. JavaScript se tedy díky tomuto prostředí dá používat i na serveru a ne jen na druhém konci, u klienta. Avšak na rozdíl např. od zmíněného PHP je v Node.js kladen důraz na vysokou škálovatelnost, tzn. schopnost obsloužit mnoho připojených klientů naráz. Pro tuto vlastnost a vysokou výkonnost je dnes Node.js velmi oblíbený pro tvorbu tzv. API serverů pro klientské single page aplikace rovněž v Javascriptu.

3.3.0.2 Shell scripty

Shell scripty jsou sada příkazů, které po spuštění zpracovává a provede shell. Význam těchto scriptů v rámci tohoto projektu jsou CI/CD a DevOps Operace, které jsou detailněji popsány v bodu ????

3.3.0.3 Firebase

Firebase je cloud computing provider nabízející služby jako hosting webových stránek, hosting databází, ale mnoho dalších jako je například machine learning, které však pro mě vzhledem k povaze projektu nejsou důležité. V tomto projektu zajišťuje všechny tyto zmíněné služby a nebýt Firebase, musel bych je všechny řešit sám, například pronájmem dalšího serveru, kde by byly databáze a podobně. Databáze, které v rámci Firebase využívám, jsou tzv. Firestore a Cloud Storage.



Popis

Firestore je NO-SQL databáze, což znamená, že jednotlivé záznamy uchovává v datovém typu Objekt, což je key-value datový typ, který slouží pro jednoduché popisování parametru daného předmětu, například člověka. Pro mé účely (ukládání záznamů o jednotlivých albech a fotografiích) je naprosto dostačující, avšak jedinou nevýhodou je, že NO-SQL databáze neumí vytvářet relace mezi jednotlivými tabulkami, tudíž, když jsem chtěl vytvořit relaci mezi albem a jeho fotografiemi, musel jsem v albu vytvořit pole s názvem connectedImages a v něm uchovávat IDs všech

těchto fotografií.

```
graph LR; dmp-bures[albums] --> albums[albums]; albums --> doc[1642016809701]; doc --> connectedImages[connectedImages]; connectedImages --> item0[0]; item0 --> imageId0["imageId: \"QJqQf0jHGZMI1kwAaxu8\""]; item0 --> imageSrc0["imageSrc: \"https://firebasestorage.googleapis.com/v0/b/dmp-bures.appspot.com/o/QJqQf0jHGZMI1kwAaxu8?alt=media&token=691db6c4-83a1-406f-96aa-15344172eba0\""]; connectedImages --> item1[1]; item1 --> imageId1["imageId: \"Z0KxJLQiBtKnnmaKdJr\""]; item1 --> imageSrc1["imageSrc: \"https://firebasestorage.googleapis.com/v0/b/dmp-bures.appspot.com/o/Z0KxJLQiBtKnnmaKdJr?alt=media&token=06f1e3bc-7b04-4c23-8534-477a00113333\""]
```

Popis

Cloud Storage je cloudové úložiště pro všechny typy souborů. Avšak vzhledem k tomu, že by pan fotograf měl nahrávat jen fotografie, ve scriptu, který řídí nahrávání nových fotografií je implementováno jednoduché ověření, zda li soubor, který se snaží nahrát, doopravdy je typu obrázku/fotky. Nicméně kdyby si to zákazník přál, jsem schopen implementovat i nahrávání souborů jiných, než fotografií.

[helper functions codeblock]

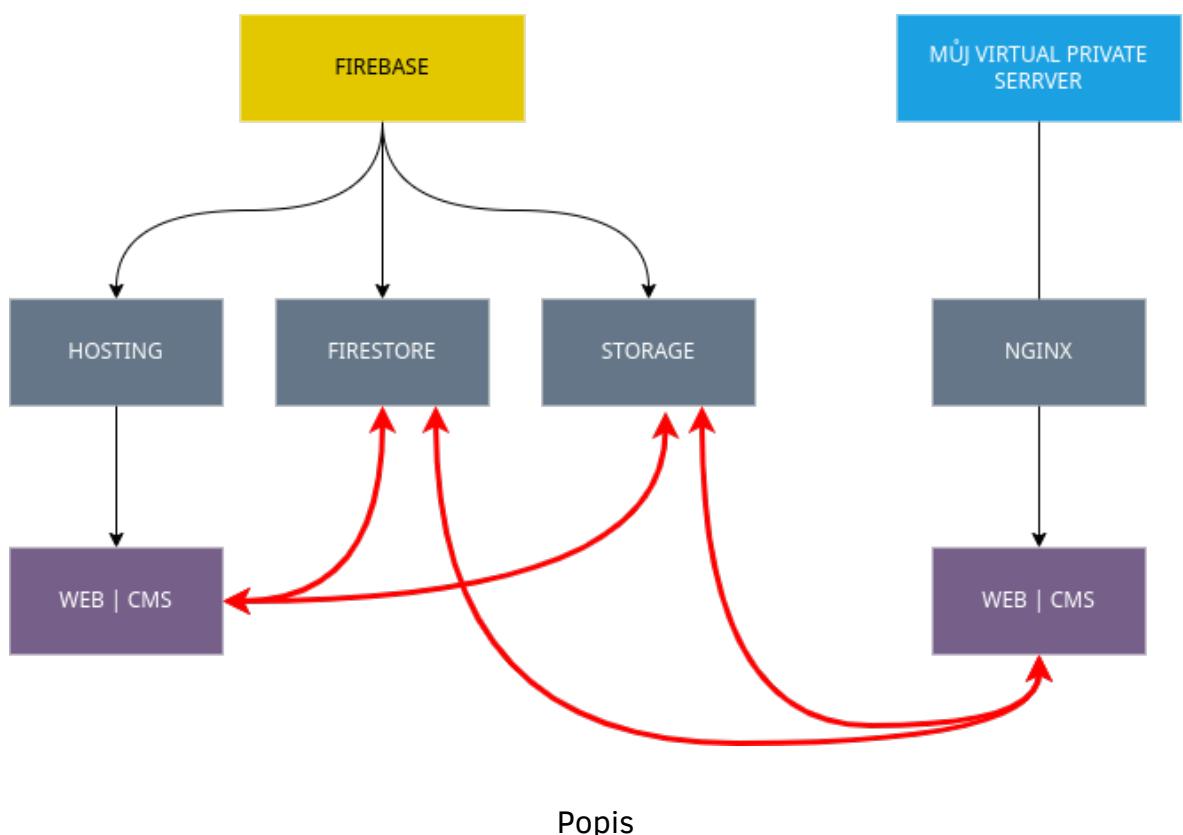
3.4 Hosting

Hosting je zřízen pomocí Firebase Hosting nejen pro jednoduchou integraci s Github repositářem a předem připravenými CI/CD pipelines, ale také díky jeho ceně. V základním tarifu, jsou všechny služby Firebase (včetně hostingu), zdarma. V rámci tohoto tarifu je však hosting limitován na 10GB úložiště, tedy projekt nesmí přesahnut stanovenou kvótou spolu s maximálním trafficem 360MB/den. Samozřejmostí je avšak SSL certifikát a možnost hostovat více než jednu stránku v rámci projektu.

Nastavení hostingu je u Firebase velmi jednoduché a díky CLI programu Firebase Tools je vývojář schopen deplounout projekt jedním jednoduchým příkazem, nebo vytvořit CI/CD script, který projekt deployne po pushnutí změn do repozitáře.

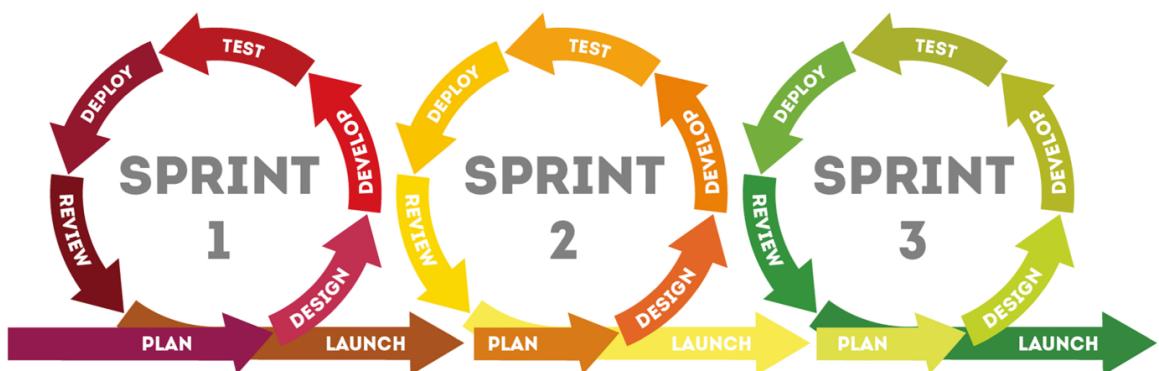
4 Struktura projektu

Projekt je rozdělen na dvě částí – komerční a školní (DMP). Co se týče části komerční, všechny služby okolo hostingu a databází bude zprostředkovávat Firebase, jelikož je to v rámci dlouhodobého hlediska nejjednodušší a hlavně nejlevnější řešení. Školní část (DMP) budu zcela spravovat já, tudíž mám pronajatý vlastní server, na kterém budu zprostředkovávat hosting a backend služby, nicméně všechna data budou poskytována a požadována z databází komerční části.



5 Průběh vývoje

Vývoj obou systémů, který započal dubnu roku 2021 jsem se snažil provádět co nejvíce agilně. To znamená, že se odehrával v takzvaných sprintech, kde jeden sprint je časový úsek pro implementaci a vyladění požadované změny. Nicméně častokrát se stalo, že jsme spolu se zákazníkem označili právě probíhající sprint za dokončený (já jsem implementoval změnu a zákazník ji označil za odpovídající), ale za pár dní mi přišel email s tím, že se zákazníkovi tato změna nakonec nelibí a chtěl by ji změnit, nicméně i když se jedná lehce o kontraproduktivní chování, je to zcela běžné i během vývoje mnohem větších a komplexnějších projektů a musí se s tímto počítat, proto je dobré, že je celý projekt modulární a není až tak složité implementovat změnu.



Popis

Vývoj obou systémů začal nejprve ve vanilla Javascriptu, bez použití jakéhokoliv frameworku, což se jevilo jako nejjednodušší řešení, jelikož jsem tehdy neuměl používat žádný framework. Čistý Javascript, je pro projekt, jako je webová stránka více než dostačující, nicméně vzhledem k modulárnosti a mnoha dalším vylepšením, které framework jako například Gatsby.js přidává, bych jej v případě, že bych měl vytvořit webovou stránku, použil. Nicméně se změnami a složitějšími funkcionalitami, které jsem postupně přidávala do administračního systému, se zdrojový kód začal jevit poměrně nepřehledným, a proto jsem se rozhodl, že jej refaktorují

do Reactu, který nejen zvýší celkovou rychlosť a performance systému, ale také přehlednosť struktury souborů a kódu, což se nakonec vyplatilo, protože nyní mám jasny přehled o tom, kde co je a jakým způsobem to funguje.

5.1 Komunikace se zákazníkem

Komunikace se zákazníkem probíhala primárně a nejvíce prostřednictvím mailů, což se mi během vývoje velmi osvědčilo, jelikož všechny informace byly sepsány přehledně na jednom místě a bylo jasné, na čem jsme se, se zákazníkem, domluvili. Proběhlo i pár telefonních hovorů, nicméně postupně jsme od nich opustili, jelikož si za chvíli ani zákazník ani já nebyl jistý, na čem jsme se vlastně domluvili, a proto jsem jej vždycky požádal, aby své požadavky sepisoval do e-mailu a vše bylo přehledné.

5.2 Organizace úkolů a změn k implementaci

Jakmile jsem dostal od zákazníka nějaký požadavek ať už ve formě e-mailu nebo telefonátu, okamžitě jsem si jej zapsal a následně přidal do úkolníčku. Těch jsem během vývoje vyzkoušel několik, ale nejvíce mi osvědčil Todoist, pro jeho přehlednosť a jednoduché ovládaní. Dá se v něm jednoduše nastavit priorita daného úkolu a ty se dají rozdělit do projektu, což je velké plus a uživatel tak nemá na hlavní stránce změř úkolů, které například ani nesouvisejí s projektem, na kterém zrovna pracuje. Jediná nevýhoda, kterou Todoist má, je absence upozornění na dobu nastaveného zpracování úkolu.

6 Webová prezentace

Webová prezentace bude panu fotografovi sloužit jako virtuální galerie fotek, aby lidé, které jeho tvorba zajímá, měli všechny jeho fotografie na "dosah ruky" a vzhledem k tehdejší koronavirové situaci je více než vhodné, aby nemuseli chodit přímo na výstavu. Samotná webová prezentace obsahuje přesně to, co obsahovala ta stávající, tedy, hlavní stránku, na které je jen galerie z alba "Hlavní", jelikož se jedná o fotografie, které se panu fotografovi líbí nejvíce a o kterých si myslí, že by mohly oslovit nejvíce lidí. Po kliknutí náhledovou fotografií se spustí galerie, která avšak nedovoluje rozkliknout informační box s více informacemi o dané fotografii. Data, které web používá k svému fungování nejsou dynamická, jelikož by to z principu fungování statické webové stránky nemělo smysl. Data se stahují z databáze ve chvíli, kdy uživatel přijde na stránku poprvé. To znamená, že pokud uživatel nahraje novou fotografií, změny na webu se projeví až ve chvíli, kdy uživatel stránku znova načte.



Úvodem
Nejnovější
Soubory
Výstavy
Biografie
Kontakt

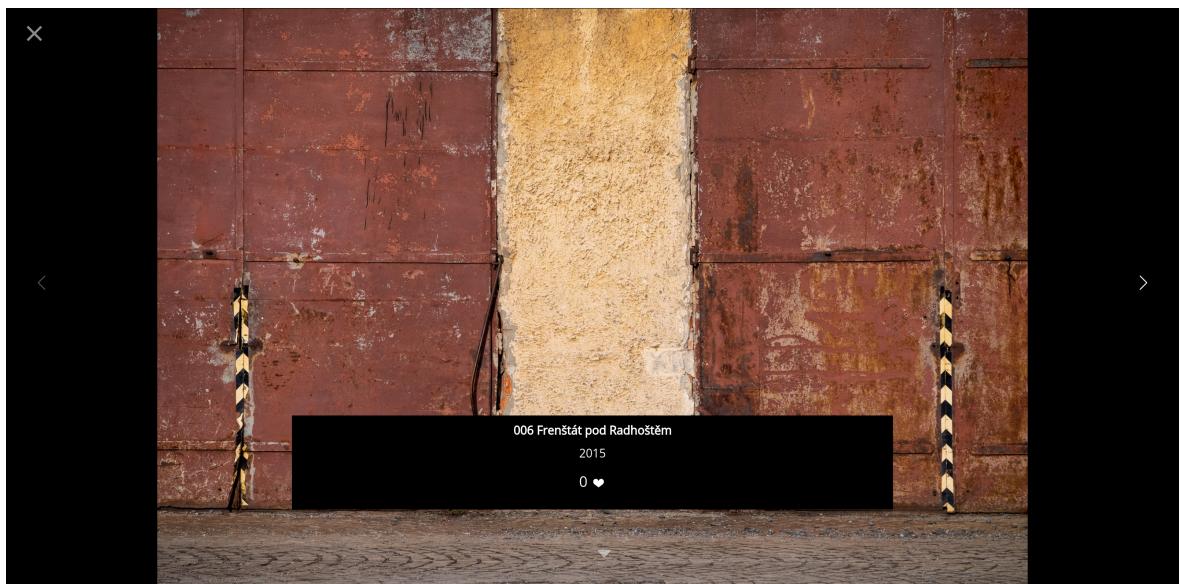


2021 © Mgr. Milan Bureš st.
Made with ❤ by Orexin.

Popis

Dále stránku "Úvodem", která slouží jako předmluva pro celý web, který má

simulovat umělecké dílo spolu s jeho uměleckými fotografiemi. Tento text je samozřejmě editovatelný v administračním systému. Odkaz na stránku alba "Nejnovější". Nejedná se o, jak by se mohlo podle názvu zdát, album s fotografiemi, které byly nahrány nedávno, ale o fotografie, které byly nedávno vyfoceny, ostatní alba totiž mohou obsahovat i fotografie, které byly vyfoceny před rokem 2000. To znamená, že se jedná o výběr fotografií, které pan fotograf nedávno vyfotil a rozhodl se, že je bude sdílet. Při vývoji této podstránky jsem musel vymyslet systém, jakým budu fotografie na webu zobrazovat, jelikož si zákazník přál, aby místo galerie, která je použita u všech ostatních alb, byla použita tabulka s náhledy fotografií. Abych tohoto docílil, spolu se správnou pagination, musel jsem pole se všemi fotografiemi rozdělit na části po 6 fotografiích a poté je v těchto částech zobrazit na webu. Po kliknutí na náhled fotografie se však otevře galerie, jako u všech ostatních alb.



Popis

Dropdown menu s odkazy na existující alba. Jelikož jsou alba dynamickými prvky v databázi, musel jsem vymyslet způsob, jakým budu vytvářet podstránky jednotlivých alb, nemohl jsem totiž ručně pro každé album vytvořit podstránku po každé, co ji uživatel administračního systému vytvoří. Implementoval jsem tedy algoritmus, který na základě id obsažené v URL linku alba vytáhne z databáze správný záznam o albu právě pomocí daného id.

[show-correct-album logic codeblock]

Stránku "Výstavy", která obsahuje seznam všech výstav, na kterých pan fotograf kdy vystavoval své fotografie. Samozřejmě se jedná o editovatelný text. "Biografie" popisuje dosavadní život pana fotografa společně s jeho fotografií.

V poslední řadě "Kontakt", kde lze najít veškeré kontaktní možnosti. Formulář, který podstránka obsahuje, bude pomocí Firebase Functions posílat email a push notifikaci panu fotografovi vždy, když mu někdo napiše email, aby se nikdy nestalo, že nebude vědět o tom, že mu někdo psal.

The screenshot shows a contact form page with a dark header featuring a logo. On the left, there's a sidebar with navigation links: Úvodem, Nejnovější, Soubory, Výstavy, Biografie, and Kontakt. The main content area has a title "Kontakt". It contains three input fields: "Vaše jméno a příjmení" (Jan Novák), "Váš E-mail" (jan-novak@priklad.cz), and "Vaše telefonní číslo" (+420 123 456 789). To the right, a note says "Odesláním souhlasíte se zpracováním Vašich osobních údajů." Below the form is a "ODESLAT" button. At the bottom, there are four social media icons with corresponding links: a phone icon for "+420 123 456 789", an envelope icon for "info@info.cz", a camera icon for "@milanbures", and a Facebook icon for "Milan Bureš".

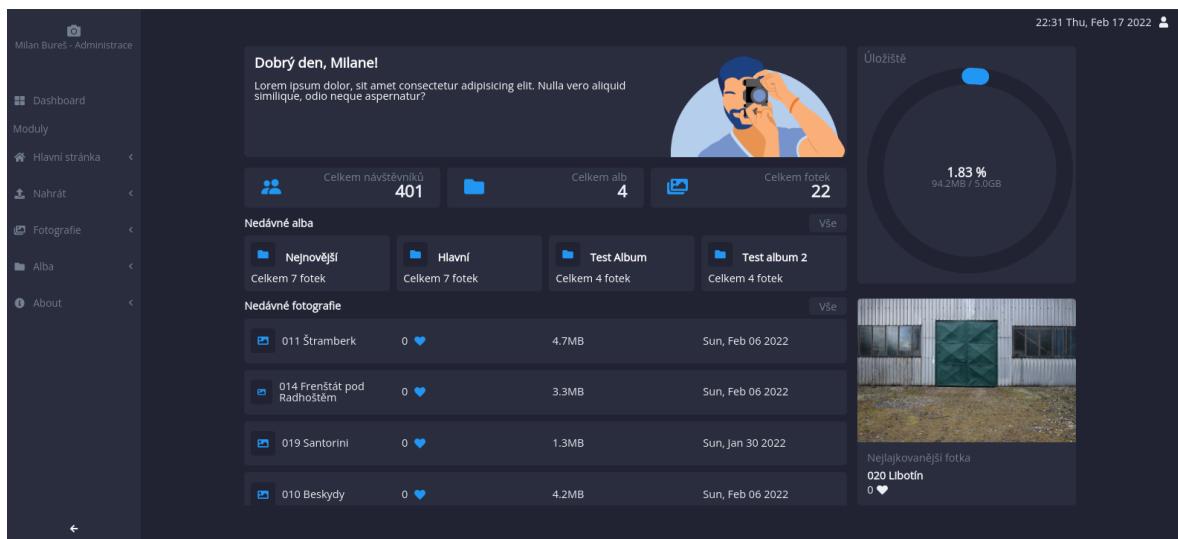
2021 © Mgr. Milan Bureš st.
Made with ❤ by Orexin.

Popis

Jelikož je webová prezentace napsána ve vanilla Javascriptu, v budoucnu mám v plánu ji přepsat pomocí frameworku Gatsby.js, který slouží pro generování statických HTML stránek za použití React.js, což bude mít za následek nejen mnohonásobně jednodušší budoucí vývoj, ale také rychlejší načítací časy.

7 Administrační systém

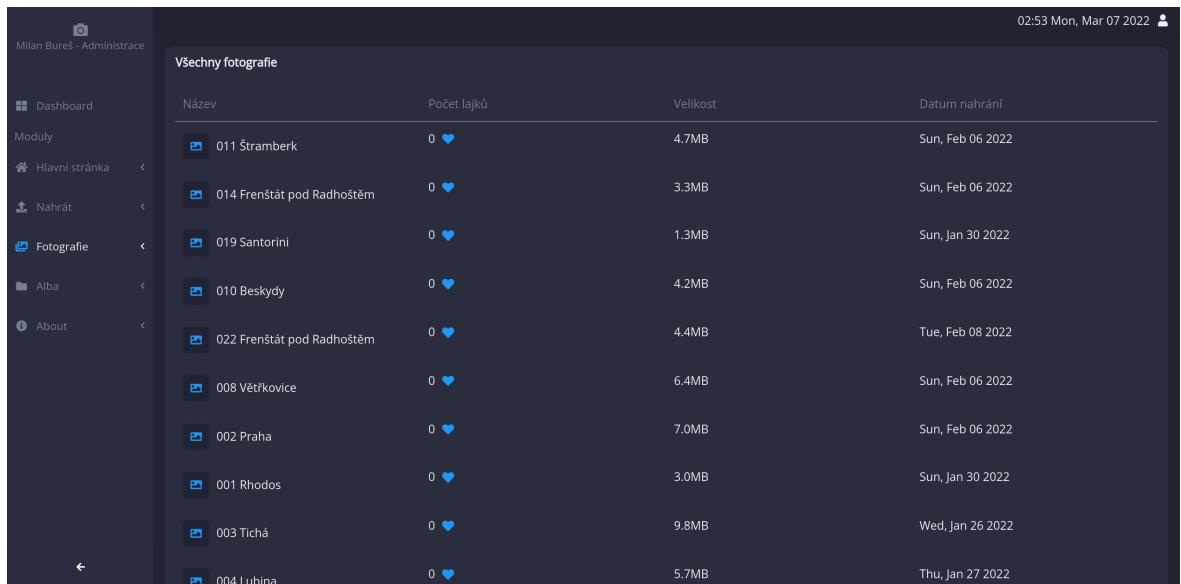
Administrační systém je systém které zprostředkovává manipulaci s daty, které se objevují na dané webové stránce. Jednoduše řečeno, díky administračnímu systému je uživatel schopen své fotografie nahrávat, mazat, upravovat jejich název a popis, stejně tak jako uvidí, která z nich je nejvíce oblíbená na základě lajků a podobně. Dále administrační systém obsahuje funkcionality pro manipulaci s alby, do kterých bude uživatel své fotografie rozdělovat. Ty samozřejmě může obdobně vytvářet, mazat a upravovat jejich fotografický obsah. V neposlední řadě systém obsahuje i informační nástěnku, kde je velmi dobře vidět například kolik fotografií je v danou chvíli nahraných na webu a v jakých albech, kolik úložného prostoru jeho fotografie zabírají a nebo třeba seznam posledních nahraných fotografií, aby je nemusel hledat.



Popis

Všechny data, se kterými uživatel pracuje, jsou zobrazována v reálném čase, což znamená, že pokud, například upraví název fotografie, tato změna se ihned projeví jak v samotném administračním systému, tak v databázi a tím pádem i na webové stránce. Díky Firebase Auth se do systému nedostane uživatel, který není

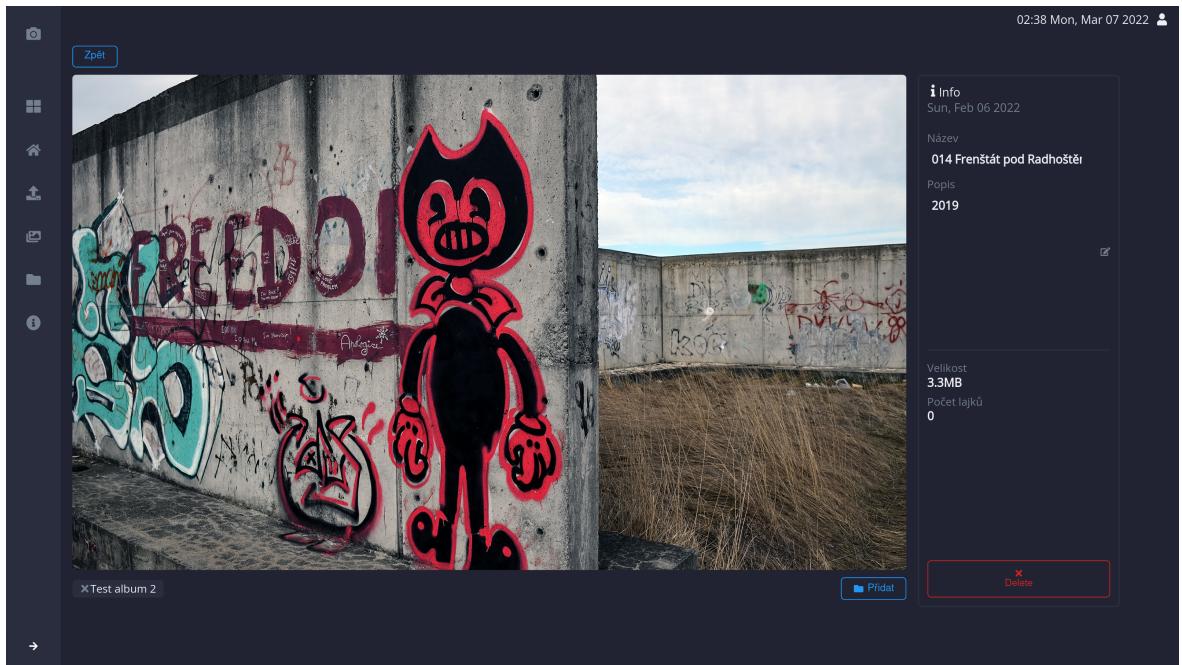
autentifikovaný, tedy přihlášený. Prozatím celý autentifikační systém funguje za pomocí emailu a hesla, jelikož web spravuje jeden člověk a není potřeba, abych implementoval například autentifikaci pomocí Google či jiného již existujícího účtu. Celá aplikace je "obalená" v auth elementu, který po úspěšném přihlášení (ověří se, zda uživatel existuje na Firebase Auth serveru) uchovává informace o přihlášeném uživateli a pokud záznam o tomto uživateli existuje, auth element jej pustí dál, jinak bude přesměrován zpět na přihlašovací obrazovku. Administrační systém obsahuje veškeré nástroje k tomu, aby uživatel mohl manipulovat s fotografiemi a alby, které nahraje a vytvoří. Po nahrání jedné či více fotografií se tyto fotografie objeví na podstránce "Fotografie" v přehledném seznamu. Kliknutím na jednotlivé fotografie se uživatel dostane na stránku pro manipulaci s jednotlivou fotografií, kde ji může přejmenovat, přidat popis, přiřadit do alba nebo ji z alba odstranit, a nebo fotografii odstranit úplně. Také lze vidět metadata fotografie, jako například kdy byla nahrána, či případně kolikrát ji návštěvníci webové prezentace udělili "lajk".



The screenshot shows a dark-themed administrative dashboard for a photo management system. On the left is a sidebar with navigation links: Dashboard, Moduly (Hlavní stránka, Nahrávání, Fotografie, Alba), and About. The main content area is titled 'Všechny fotografie' (All photos). It displays a table with the following columns: Název (Name), Počet lajků (Number of likes), Velikost (Size), and Datum nahrání (Upload date). The table lists ten photos:

Název	Počet lajků	Velikost	Datum nahrání
011 Štramberk	0 ❤️	4.7MB	Sun, Feb 06 2022
014 Frenštát pod Radhoštěm	0 ❤️	3.3MB	Sun, Feb 06 2022
019 Santorini	0 ❤️	1.3MB	Sun, Jan 30 2022
010 Beskydy	0 ❤️	4.2MB	Sun, Feb 06 2022
022 Frenštát pod Radhoštěm	0 ❤️	4.4MB	Tue, Feb 08 2022
008 Větřkovice	0 ❤️	6.4MB	Sun, Feb 06 2022
002 Praha	0 ❤️	7.0MB	Sun, Feb 06 2022
001 Rhodos	0 ❤️	3.0MB	Sun, Jan 30 2022
003 Tichá	0 ❤️	9.8MB	Wed, Jan 26 2022
004 Lubina	0 ❤️	5.7MB	Thu, Jan 27 2022

Popis



Popis

A screenshot of a dark-themed photo management application. On the left is a sidebar with navigation links: "Dashboard", "Moduly", "Hlavní stránka", "Nahrát", "Fotografie", "Alba", and "About". The main area shows three album thumbnails:

- Thumbnail 1: "Hlavní Celkem: 7 položek" (Main Total: 7 items)
- Thumbnail 2: "Test Album Celkem: 4 položek" (Test Album Total: 4 items)
- Thumbnail 3: "Test album 2 Celkem: 4 položek" (Test album 2 Total: 4 items)

At the top right, the date and time are shown: "02:52 Mon, Mar 07 2022".

Popis

8 VPS - Virtual Private Server

Pro potřeby práce jsem si pronajal a nakonfiguroval vlastní server, který kopíruje veškeré funkcionality Firebase. Zajišťuje hosting pomocí softwarového webového serveru nginx, díky čemuž si na adrese VPS může zobrazit webovou stránku i modifikovat její obsah přes administrační systém. Kromě toho na VPS běží i Node.js backend server, díky kterému bude moct uživatel této instance projektu odesílat emaily panu fotografu bez použití email serveru třetí strany. De-facto se jedná o druhou instanci komerčního projektu pro pana fotografa.

8.1 VPS provideři

Na trhu je mnoho cloud computing providerů, nabízejích virtuální počítače v nejrůznějších početně výkonnostních kategoriích. Mezi nejznámější patří Digital Ocean, Linode, Hostinger nebo Amazon Cloud, kterému jsem se chtěl ale zámerně vyhnout, jelikož se mi vůbec nelibí prostředí, ve kterém uživatel své VMs spravuje a jejich ceník je poměrně nepřehledný.

Provider	Cena/měsíc	Disk	RAM	Transfer (\$/GB)
Digital Ocean	\$5	25GB	1GB	\$0.01
Hostinger	\$3.95	20GB	1GB	Do 1TB/mešíc zdarma
Linode	\$5	25GB	1GB	Do 1TB/měsíc zdarma
A2	\$8.99	150GB	1GB	Do 2TB/měsíc zdarma

Já jsem se rozhodl pro DigitalOcean, jelikož s nimi mám již předešlou zkušenost a jejich prostředí pro správu virtuálních počítačů mi ze všech konkurenčních provi-

derů přijde nejvíce intuitivní a přehledné, spolu s ceníkem, který neobsahuje žádné lžsti. Jejich cena není ze všech uvedených na listu nejlevnější, ale jedná se o kvalitní službu, která je této ceně adekvátní.

8.2 Konfigurace VPS

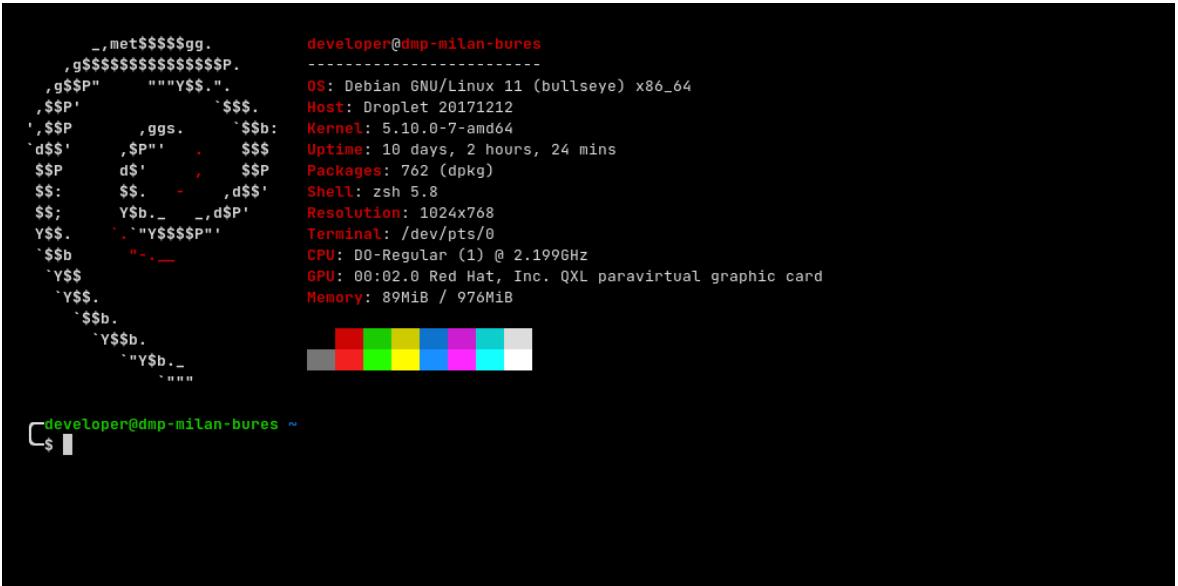
8.2.1 Operační systém GNU/Linux

V dnešní době je jako serverový operační systém nejvíce rozšířený GNU/Linux pro jeho minimalističnost a poměrně jednoduchou konfiguraci. Podle hostingtribunal.com až 96 procent serverů z 1 milionu celosvětově nejpoužívanějších serverů používá Linux jako svůj operační systém a proto jsem se i já rozhodl použít Linux a nejen z toho důvodu, že jej dennodenně používám.

8.2.1.1 Debian 11

Původně jsem na svůj VPS chtěl nainstalovat Arch Linux, jelikož se jedná o Linuxovou distribuci, kterou dennodenně používám, nicméně má jednu velkou nevýhodu - nemá vlastní instalátor (i když nyní již existuje instalační script), což mě osobně při instalaci v domácích lokálních podmínkách vůbec nevadilo, ale v případě instalování Arche na VPS by to znamenalo, že bych musel mít vlastnoručně přizpůsobené ISO, které bych použil k instalaci, jelikož DigitalOcean nenabízí Arch k nainstalování na jejich VPS, které nazývají Droplets. Volba tedy padla na Debian 11, jelikož je v jeho minimalistické instalaci poměrně "čistý" a tudíž neobsahuje zbytečné programy a podobně (bloatware). Vzhledem k tomu, že tento "počítač" bude sloužit jako server, není potřeba a je až zbytečné instalovat jakékoli desktopové prostředí GUI (Graphical User Interface), jelikož by to mělo nejen za následek zvýšení požadavků na výpočetní výkon serveru, což je nežádoucí, jelikož chceme, aby operační systém serveru zabíral co nejméně prostoru a prostředků, a zároveň celá konfigurace systému je proveditelná jen za pomoci příkazové řádky CLI (Command Line Interface). Na obrázku pod lze vidět, jak CLI-only Debian v idle stavu (stavu, kdy neprovádí žádné operace) zabírá na paměti RAM jen 89MB z dostupných 1GB, tedy necelých

10 procent.



```
_,met$$$$$gg.          developer@dmp-milan-bures
,g$$$$$$$$$$$$$$$$$P.  -----
,$$P"    ""Y$$.".      OS: Debian GNU/Linux 11 (bullseye) x86_64
,$$P'      '$$$.        Host: Droplet 20171212
,$$P      ,ggs.        Kernel: 5.10.0-7-amd64
,$$P"    '$$b:        Uptime: 10 days, 2 hours, 24 mins
,$$P      '$$b:        Packages: 762 (dpkg)
,$$P      d$'.        Shell: zsh 5.8
,$$:      $$..        Resolution: 1024x768
,$$;      Y$b._ ,d$P'  Terminal: /dev/pts/0
Y$$.    .`"Y$$$P"'.   CPU: D0-Regular (1) @ 2.199GHz
`$$b    "-.-
`Y$$
`Y$$.      '$$b.        GPU: 00:02.0 Red Hat, Inc. QXL paravirtual graphic card
`$$b.
`Y$$b.
`"Y$b._  Memory: 89MiB / 976MiB
`""

developer@dmp-milan-bures ~
```

Popis

8.2.2 Vytvoření developer uživatele

Tento uživatel bude využíván jako správce celého vývojového i produkčního prostředí a bude mít skoro stejná práva jako root.

```
useradd -m developer
passwd developer
```

Aby developer mohl naplno využívat systém, je potřeba, aby mohl používat sudo , které bylo díky Digital Ocean předinstalováno. Přidáme jej tedy do sudoers skupiny.

```
usermod -aG sudo developer
```

A ověříme, že vše správně dopadlo:

```
su developer
sudo -l
[sudo]":
User developer may run the following commands on dmp-milan-bures:
(ALL : ALL) ALL
```

8.2.3 Konfigurace prostředí

8.2.3.1 Shell

Vzhledem k tomu, že během práce s VPS používám git, který funguje na takzvaných větvích, je dobré vědět, se kterou větví právě pracuji. Proto nainstalují jiný shell, který mi dovolí si jednoduše upravit příkazovou řádku.

```
sudo apt-get install zsh
```

Zsh neboli Z-shell je mocný příkazový shell který odemyká mnoho možností ať už po stránce přizpůsobení vzhedu ale také vylepšení funkcí. V kombinaci s open-source projektem oh-my-zsh jsem schopen změnit nejen vzhled celé příkazové řádky, ale také přidat pluginy jako například git, který mi, pokud se v aktuálním adresáři nachází .git soubor(soubor, který inicializuje složku jako git repozitář), zobrazí aktuálně zvolenou větev. Oh-my-zsh nainstaluje pomocí nástroje wget přímo z oh-my-zsh repozitáře na GitHubu, který poté spustí instalační script.

```
wget https://raw.github.com/ohmyzsh/ohmyzsh/master/tools/install.sh
sh install.sh
```

8.2.3.2 Git a verzovací systém

Git je open-source nástroj pro správu verzí zdrojového kódu anebo jeho součástí vytvořený Linusem Torvaldsem. Jednoduše řečeno jedná se o software, který sleduje změny v souborech, aby pak měl uživatel/vývojář větší přehled o tom, jaké změny provedl. V současnosti se git nejvíce používá pro kolaboraci mezi vývojáři na open-source ale i closed source projektech, jelikož git samotný je open-source, tudíž jej může kdokoliv využívat na svém interním serveru. Vzhledem k tomu, že z

VPS nebudu pushovat žádné změny do remote repositáře, nebylo potřeba provádět žádnou pokročilejší konfiguraci a git stačí jen nainstalovat.

```
sudo apt-get install git
```

Jediné, co jsem nakonfigurovat musel je zachování přihlašovacích údajů tak, aby se git nemusel pokaždé při každém pullu dotazovat na přihlašovací údaje k remote repository serveru, v mém případě ke Githubu.

```
git config --global credential.helper store
```

8.2.3.3 Node.js a NPM

Javascript samotný je spustitelný jen v prohlížeči, kde většinou plní funkci logického "mozku" webové stránky a provádí logické operace. Díky tomu avšak nemá moc jiného využití, což Node.js mění. Node.js je Javascript runtime, díky kterému můžeme spustit Javascriptový kód i mimo prohlížeč, jako například v příkazové řádce jako ostatní scriptovací ale i programovací jazyky, jako například Python nebo Java.

Umožňuje nám vyvíjet API servery pro webové stránky i webové aplikace. Jelikož je webová prezentace i administrační systém serverless, není sám o sobě Node.js potřeba pro správnou funkčnost systému, ale vzhledem k faktu, že používím third-party knihovny a pluginy jako například webpack, musím Node.js nainstalovat. Tyto knihovny a další podpůrné programy lze nainstalovat pomocí takzvaného Node Package Manageru (Správce balíčků pro Node.js), neboli NPM.

```
sudo apt-get install nodejs npm
```

Zjistíme verzi Node.js:

```
node -v  
[node]: v12.22.5
```

A jelikož verze 12 není nejnovější, musím aktualizovat na nejnovější stable verzi, jinak by se totiž mohlo stát, že nějaká z dependencies, které v projektu používám, nebude se verzí 12 spolupracovat.

```
sudo npm i -g node  
sudo npm i -g npm
```

A opět zkontrolujeme verzi Node.js:

```
node -v  
[node]: v16.14.0
```

8.2.3.4 Nginx

Nginx je open source softwarový webový server a load balancer, který se zaměřuje hlavně na co nejmenší konsumpcí paměti a výkonu a skvěle zvládá velký nápor připojených zařízení. Nainstalujeme jej z Debian repositářů.

```
sudo apt-get install nginx
```

A spustím pomocí systemctl.

```
sudo systemctl start nginx
```

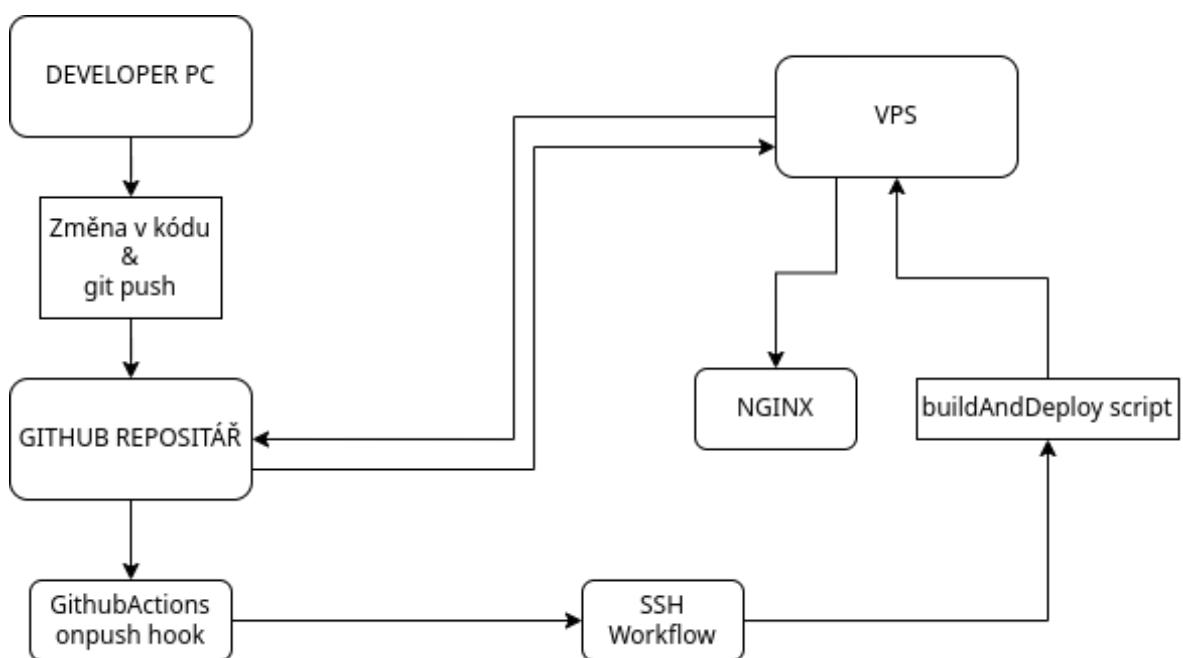
Nyní po úspěšném zapnutí nginx se VPS chová jako webový server a pokud v prohlížeči přejdu na adresu <http://159.65.112.226/> (IP adresa VPS), zobrazí se přivítací obrazovka nginx, což indikuje, že je všechno správně nakonfigurováno a funkční.

8.3 Konfigurace CI/CD

CI (Continuous Integration) a CD (Continuous Delivery) je proces, který umožňuje nepřetržité dodávání aktuální verze projektu k testování či produkčnímu spuštění. V praxi to znamená, že pokud tým vývojářů provede změnu a uloží ji do remote repositáře (místo pro ukládání zdrojových kódů v cloudu), tato změna se automaticky projeví (deployne) na produční server (server, na kterém je spuštěna aktuální verze projektu a je poskytována uživatelům z venčí). Po té, co provedu změnu v zdrojovém kódu webové prezentace nebo administračního systému a tuto změnu nahraji pomocí verzovacího systému git do Github repozitáře, je potřeba, aby se

tato změna projevila na produkčním serveru. K tomu využiji shell scripty a Github actions hooky.

Celá operace probíhá tak, že jakmile vývojář (v tomto případě já) provede změny ve zdrojovém kódu a pushne je do remote repositáře, Github Action detekují push a spustí interní script napsaný v jazyce YAML, který se pomocí protokolu SSH připojí s předem připravenými přihlašovacími údaji přihlásí a VPS, kde spustí buildAndDeploy script. Ten pomocí gitu pullne změny na lokální úložiště a spustí build scripty, což znamená, že bundler překompiluje zdrojové soubory projektu tak, aby byly optimalizované pro použití v produkčním prostředí a aby je dokázal softwarový webový server nabízet uživatelům z vnější sítě.



Popis

8.3.1 Naklonování projektu na lokální úložiště

Jelikož není nejlepším řešením mít inicializovaný lokální klon git repozitáře přímo ve složce, ze které nabízím webové stránky do vnější sítě, vytvořím složku, ve které se bude klon repozitáře nacházet a projekt do ní naklonuji.

```
mkdir ~/files  
cd files  
git clone https://github.com/[repository].git
```

Nyní mám na systému přesnou kopii projektu a proto vytvořím symlink, který uložím do složky, kterou nginx defaultně používá pro soubory nabízených webových stránek.

```
ln -s ~/files/dmp-bures/website/dist /var/www/dmpbures.cz  
ln -s ~/files/dmp-bures/dashboard/build /var/www/admin.dmpbures.cz
```

8.3.2 Nginx konfigurace pro web i administrační systém

Aby byl web i administrační systém přístupný z vnější sítě, je pro každý z těchto "webů" vytvořit vlastní konfigurační soubor aby nginx věděl, kde se nacházejí jednotlivé zdrojové soubory a pod jakou adresou mají být přístupné. Pro web i administrační systém vytvořím konfigurační soubor v adresáři "/etc/nginx/sites-available".

/etc/nginx/sites-available/dmpbures.cz

```
server {  
    listen 80;  
    listen [::]:80;  
    root /var/www/dmpbures.cz;  
    index index.html;  
    server_name dmpbures.cz www.dmpbures.cz;  
    location / {  
        try_files $uri $uri/ =404;  
    }  
}
```

/etc/nginx/sites-available/admin.dmpbures.cz

```
server {  
    listen 81;  
    listen [::]:81;  
    root /var/www/admin.dmpbures.cz;  
    index index.html;  
    server_name admin.dmpbures.cz www.admin.dmpbures.cz;  
    location / {  
        try_files $uri $uri/ =404;  
    }  
}
```

Struktura konfiguračních souborů je poměrně jednoduchá a jasná:

1. **listen** - Na jakém portu web poběží. Všimněte si prosím, že se porty webu a CMS liší, aby se navzájem nekryli a uživatel tak mohl přistoupit k oboum webům.
2. **root** - Složka ve které se nachází zdrojové soubory dané stránky
3. **index** - Soubor který slouží jako vstupní HTML dokument (zobrazí se jako první)
4. **server name** - Název serveru, pod jakým je web přístupný. V lokální síti by to bylo pod specifikovaným názvem, ve vnější v případě absence DNS záznamu pod IP VPS.
5. **location** - Udává lokaci dalších souborů, jako například 404 stránky (která se zobrazuje, pokud vyhledávaná stránka neexistuje)

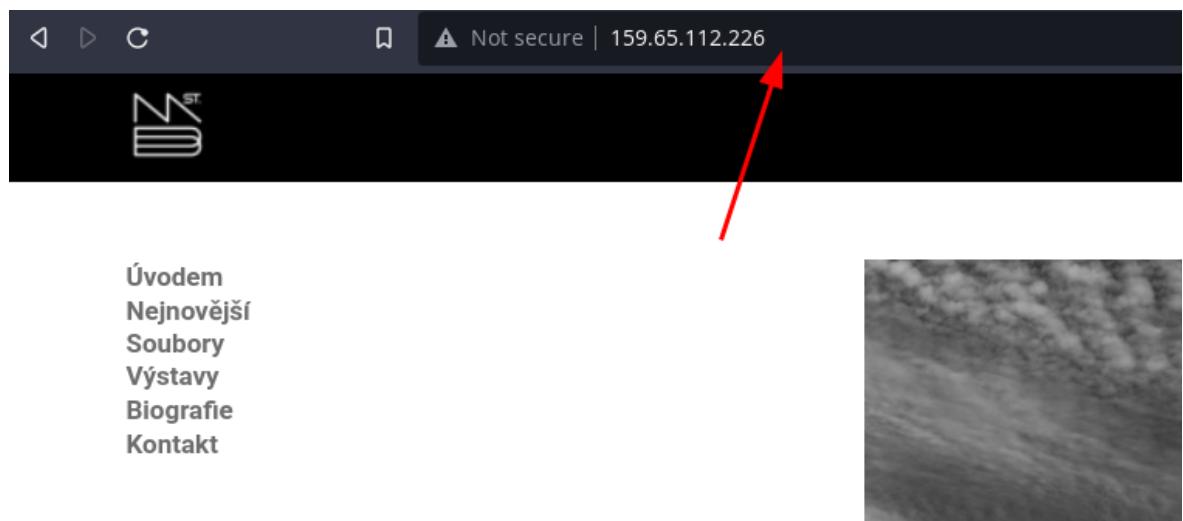
Ve složce /etc/nginx/sites-available se však nachází jen konfigurační soubory webů, které jsou podle názvu složky k dispozici, nicméně nejsou aktivně nabízeny uživatelům z vnější sítě. Abych toho dosáhl, musím vytvořit kopii konfiguračních souborů ve složce /etc/nginx/sites-enabled, například pomocí symlinků.

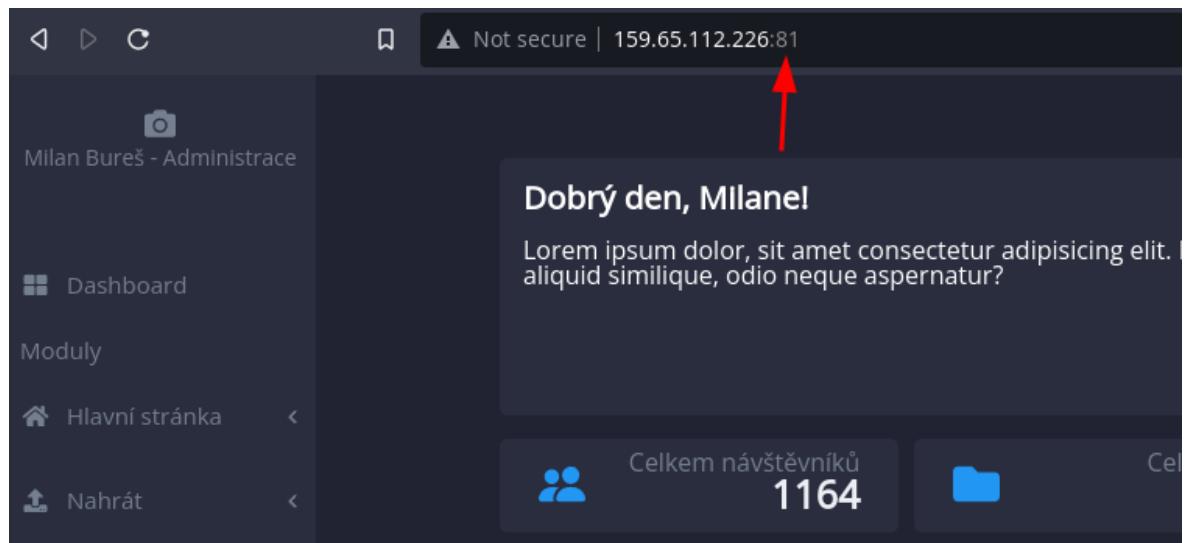
```
ln -s /etc/nginx/sites-available/dmpbures.cz  
/etc/nginx/sites-enabled/  
ln -s /etc/nginx/sites-available/admin.dmpbures.cz  
/etc/nginx/sites-enabled/
```

Nyní mám nakonfigurovaný nginx i weby, které má nabízet uživatelům z vnější sítě. Aby vše začalo fungovat, nginx restartuji a ověřím konfiguraci.

```
nginx -t  
sudo systemctl restart nginx  
systemctl status nginx
```

Po spuštění prohlížeče a vyhledání adresy <http://159.65.112.226> provede prohlížeč požadavek na můj nyní nakonfigurovaný webový server, kde jej nginx zachytí, přečte si adresu, kterou prohlížeč požaduje a pošle zpět odpovídající html soubor(y), v tomto případě, jelikož je defualtní port 80, webovou prezentací. Kdyby uživatel vyhledal adresu <http://159.65.112.226:81>, webový server by odpověděl a odesla zpět soubory administračního systému a uživatel by tak mohl editovat obsah.





Popis

9 Závěr

Během vývoje tohoto projektu jsem se naučil mnoho nových věcí, které věřím, že mě v mém profesionálním životě posunuly dál, a jsem rád, že jsem si jako dlouhodobou maturitní práci vybral právě tento projekt. Mrzí mne však, že jsem začal pozdě s vývojem pomocí frameworku React.js, jelikož bych měl vývoj snazší a neztratil bych čas vyvíjením administračního systému ve vanilla Javascriptu a tím pádem bych se mohl soustředit na důležitější věci, než je opětovné vyvíjení systému, který jsem již jednou vytvořil.

10 Seznam použitého softwaru

1. **Microsoft Visual Studio Code** - IDE použité k vývoji celého projektu.
2. **LaTeX** - Markup jazyk pro psaní prací ve velké typografické kvalitě pomocí profesionály předdefinovaných typografických stylů.
3. **Gimp** - Open-source rastrový editor pro editaci grafických objektů.
4. **Typora** - Textový editor pro textové soubory typu Markdown pro jednodušší práci s nimi.
5. **Git** - Verzovací systém .
6. **Github** - Server pro ukládání remote git repozitářů.

11 Seznam obrázků

12 Citace