Dlouhodobá maturitní práce

Školní rok: 2021/2022

Autor: Jiří Vala

Název: Fotografický web a kompletní administrační systém

Zadání v bodech:

1. Návrh a koncepce webu - shrnutí požadavků zákazníka návrh řešení
2. Použité platformy, technologie a software
3. Vývoj webové stránky
4. Vývoj administračního systému (dále jen "CMS")
5. Hosting

Vymezení pojmů

- *API*
- *SOMETHING*

Návrh a koncepce webu

Požadavky zákazníka

Pan Milan Bureš (dále jen "zákazník") požadoval řešení fotografického webu s administračním sysystémem za použití nejmodernějších technologií, jelikož jeho dosavadní web byl vytvořen pomocí technologií z minulého desetiletí, a proto se rozhodl pro kompletní vylepšení. Požadoval, aby si web zachoval ty samé designové a funkční prvky a podle toho se také vývoj odvíjel a byl kladen velký důraz na to, aby se web podobal co nejvíce tomu původnímu, ale zároveň aby splňoval pravidla dobrého webu.

Použité platformy, technologie a software

Pro vývoj jsem jako hlavni programovací a scriptovací jazyk využil Javascript, jelikož je masivně adaptovaný snad všemi prohllížeči a je pomerně jednoduchý, což je obrovskou výhodou během vývoje komplexní webové aplikace jako je CMS. Pro webovou stránku jsem zvolil čistý vanilla javascript (bez použití frameworků), což se v prvních fázích vývoje jevilo jako optimálné řešenív poměru komplexnosti a funkcionalit, ale jak projekt nabíral na složitosti mě začalo poměrně mrzet, že jsem se nerozhodl využít frameworku jako je například Gatsby.js, který by mi vývoij masivně usnadnil, ale v budoucnu, jelikož je tento projekt komerční, mám v plánu webovou stránku přepsat právě za pomocí Gatsby.js.

CMS jsem původně také napsal ve vanilla javascriptu, ale ve chvíli, kdy jsem měl začít přidávat komplexnější a složitější funkce jsem se rozhodl, že celé CMS přepíšu pomocí frameworku React.js

Github a git

Už na začátku vývoje bylo jasné, že bude potřeba využití verzovacího systému a mít cloudové úložiště, kde budu moct kód ukládat. Jasná volba padla na git a github, jelikož se jedná o nejvíc rozšířené a neosvědčenější rešení pro mé potřeby verzování.

Visual Studio Code

Jako hlavní editor byl využit vscode, jelikož se jedná o velice povedený editor s mnoha funkcemi jako je například integrace předem zmíněného gitu, takže je verzování velice pohodlné a vývojář přesně vidí, kde co commituje a kam.

Firebase Tools

Firebase je cloud functions provider od Google. Pro zákazníka zprostředkovává jak hosting, tak databázi pro data a metadata fotografií a alb, ale také samotné úložistě fotek.

Linux část DMP

```
rootPassword: Milan0Bures
hostname: dmp-milan-bures
```

Digital Ocean VPS

Digital Ocean je cloud provider a jelikož mi Google Cloud nechtěl verifikovat platební kartu, musel jsem zvolit jiného providera a nakonec volba padla na Digital Ocean.

Kromě Droplets (takto nazývají své VMs) nabízejí mnoho dalších služeb jako Container based apps (Docker, Kubernetes) nebo jen čistý hosting frontend statických webů, ale i backend aplikace postavené například na node.js.

Během vytváření Dropletu jsem měl možnost vybrat si z několika Linux distribucí a ikdyž jsem chtěl zvolit arch linux jelikož jej používám na svém počítači, zvolil jsem nakonec možnost Debianu 11, jelikož jsme jej nejen používali ve škole, ale nemusel jsem jej instalovat. Kdybych si zvolil arch, tak ikdyž již existuje instalační script, musel bych mnoho věcí konfigutovat a nastavovat, zatímco Debian byl hned pár minut po vytvoření VPS ssh-ready a mohl jsem se připojit a začít nastavovat developer uživatele, nainstalovat potřebné balíčky pro zfungování development i production prostředí (git, node.js, nginx a další) a pomalu začít se spouštěním projektu.

Vytváření Developer Uživatele

Tento uživatel bude využíván jako správce celého dev i prod enviromentu a bude mít skoro stejná práva jako `root`.

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

```
username: developer
password: dmpDeveloper123
```

Aby developer mohl naplno využívat systém, je potřeba, aby mohl používat `sudo`, které díky Digital Ocean bylo preinstalled. Přídáme jej tedy do `sudoers` groupy.

```
usermod -aG sudo developer
```

... a vyzkoušíme, zda-li má developer opravdu přístup k `sudo`:

```
su developer
sudo
usage: sudo -h | -K | -k | -V
usage: sudo -v [-AknS] [-g group] [-h host] [-p prompt] [-u user]
usage: sudo -l [-AknS] [-g group] [-h host] [-p prompt] [-U user] [-u user]
[command]
usage: sudo [-AbEHknPS] [-r role] [-t type] [-C num] [-D directory] [-g group] [-
h host] [-p prompt] [-R directory] [-T
            timeout] [-u user] [VAR=value] [-i|-s] [<command>]
usage: sudo -e [-AknS] [-r role] [-t type] [-C num] [-D directory] [-g group] [-h
host] [-p prompt] [-R directory] [-T
            timeout] [-u user] file ...
```

Konfigurace developer enviromentu

Shell

Aby se mi během celé konfigurace dobře pracovalo, rozhodl jsem se, že si nastavím systém tak, aby fungoval a obsahoval to, co denně používám. Například nainstaloval jsem jiný `shell`, jelikož ten původní byl defaltní `bash` ale používám `zsh`.

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

Zsh sám o sobě je velice powerful a odemyká mnoho možností, ale v kombinaci s projektem *oh-my-zsh* se síla zsh posouvá úplně jinde. Instalaci provedeme pomoci inbuilt toolu `wget` přímo z *oh-my-zsh* repozitáře.

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

Plugin pro git je předinstalovaný, a díky oh-my-zsh se sh prompt změní a uživatel tak například vidí, na které git branchi právě je.


Nginx Přivítací obrazovka

Git

Samozřejmě, že pro to, abych si mohl naklonovat zdrojový kód projektu a případně i nějaké další balíčky pro systém, budu potřebovat git. Jedná se o version control systém a je určený pro správu remote repositářů. Vzhledem k tomu, že tento systém nebude provádět nějaké změny, které pak bude zpět do repozitáře *pushovat*, není třeba provádět konfiguraci gitu a stačí ho jej stáhnout a

naklonovat repozitář, nicméně samotné klonování budu provádět až ve chvíli, kdy budu mít vše ostatní nastavené.

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

Poté jsem musel git nakonfigurovat, aby se mě vždy, při provádění nějaké git operace neptal na přihlašovací údaje.

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

Tento příkaz znamená, že si git uloží mé přihlašovací údaje, což by mohlo být poměrně nebezpečné a bylo by lepší použít SSH a ne HTTP.

Node.js a npm

Node.js je runtime pro javascript, což je scriptovací (a v dnešní době by se již dal považovat i za programovací) jazyk, který jsem použil pro webovou stránku, tak i administrační systém. Umožňuje vytvářet local virtuální servery a používá se jako pro backend pro weby a webové aplikace, například pro vývoj API. Jelikož je administrační systém serverless, není node.js sám o sobě potřeba na backend, ale jelikož v obou projektech využívám third-party knihovny jako například webpack, musím jej nainstalovat. Tyto knihovny lze nainstalovat pomocí takzvaného **N**ode **P**ackage **M**anageru, neboli NPM.

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

Zjistíme verzi Node.js:

```
node -v
$# v12.22.5
```

A jelikož verze 12 není nejnovější, musíme node aktualizovat na node@latest.

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

A pro zjištění aktuální verze opět použijeme přepínáč `-v` :

```
node -v
$# v16.14.0
```

nginx

Nginx je open source virtuální webový server a load balancer, který se zaměřuje hlavně na co nejmenší konsumpci paměti a výkonu a skvěle zvládá velký nápor připojených zařízení.

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

Nginx zapneme

```
sudo systemctl start nginx
```

a když otevřeme v prohlížeči http://159.65.112.226/ (IP adresa VPS), jde vidět že je nginx zapnutý a funkční, jelikož lze vidět přivítací nginx obrazovku.

 Nginx Přivítací obrazovka

```
cd /usr/share/nginx/html
ls -la
drwxr-xr-x 2 root root 4.0K Feb  8 18:50 .
drwxr-xr-x 4 root root 4.0K Feb  8 18:50 ..
-rw-r--r-- 1 root root  612 Apr 21  2020 index.html
```

CI/CD scripty

1. Vytvoření Node.js API které pomocí `fs` bude spuštět pullonpush.sh script
2. pullonpush.sh