Univerzita Hradec Králové

Pedagogická Fakulta

Diplomová práce

2021 Vladimír Macháček

Univerzita Hradec Králové

Pedagogická Fakulta

Katedra aplikované kybernetiky

Komparace technologií pro výuku tvorby webových stránek

Diplomová práce

Autor: Vladimír Macháček

Studijní program: M7503 / Učitelství pro základní školy (2. stupeň)

Studijní obor: 503T072 / Učitelství pro 2. stupeň základních škol – informatika
503T / Učitelství pro 2. stupeň základních škol – společný základ
503T075 / Učitelství pro 2. stupeň základních škol – tělesná výchova

Vedoucí práce: Ing. Petr Voborník, Ph.D.

Oponent: Mgr. Radek Němec, Ph.D.

Univerzita Hradec Králové
Pedagogická fakulta

Zadání diplomové práce

Autor:	Vladimír Macháček
Studium:	P16P0337
Studijní program:	M7503 / Učitelství pro základní školy (2. stupeň)
Studijní obor:	503T072 / Učitelství pro 2. stupeň základních škol – informatika 7503T / Učitelství pro 2. stupeň základních škol – společný základ 7503T075 / Učitelství pro 2. stupeň základních škol – tělesná výchova
Název diplomové práce:	**Komparace technologií pro výuku tvorby webových stránek**
Název diplomové práce AJ:	Comparison of technologies for teaching web development

Cíl, metody, literatura, předpoklady:

Diplomová práce na Pedagogické fakultě Univerzity Hradec Králové. Vedoucí diplomové práce Petr Voborník. Diplomová práce monitoruje současnou situaci v oblasti tvorby webových stránek. Popisuje a porovnává technologie, které je vhodné do výuky tvorby webových stránek zařadit. Vysvětluje, proč jsou dané technologie vhodné pro výuku a uvádí příklady využití.

Garantující pracoviště:	Katedra aplikované kybernetiky, Pedagogická fakulta
Vedoucí práce:	Ing. Petr Voborník, Ph.D.
Oponent:	Mgr. Radek Němec, Ph.D.
Datum zadání závěrečné práce:	9.4.2019

Prohlášení

Prohlašuji, že jsem tuto diplomovou práci vypracoval samostatně s odbornou pomocí vedoucího práce Ing. Petrem Voborníkem, Ph.D. V práci jsem uvedl všechny použité prameny, literární a odborné zdroje.

Ve Skutči dne 13.7.2021 ……..……

Podpis

Prohlášení

Prohlašuji, že diplomová práce je uložena v souladu s rektorským výnosem č. 13/2017 (Řád pro nakládání s bakalářskými, diplomovými, rigorózními, dizertačními a habilitačními pracemi na UHK).

Ve Skutči dne 13.7.2021 ……..……

podpis

Poděkování

Chtěl bych poděkovat mému vedoucímu práce panu Ing. Petru Voborníkovi, Ph.D. za usměrnění tématu diplomové práce a veškeré odborné konzultace poskytnuté během jejího vytváření. Dále bych chtěl poděkovat všem pedagogům, kteří mi vyplnili dotazník do výzkumné části. Děkuji také své rodině, že mě podporovali po celou dobu mého nekonečného studia.

Anotace

VLADIMÍR, M. *Komparace technologií pro výuku tvorby webových stránek.* Diplomová práce, Univerzita Hradec Králové, Hradec Králové, 2021.

Diplomová práce monitoruje současnou situaci v oblasti tvorby webových stránek. Popisuje a porovnává technologie, které je vhodné do výuky tvorby webových stránek zařadit. Vysvětluje, proč jsou dané technologie vhodné pro výuku a uvádí příklady využití. Porovnává poměr vyučovaných technologií ve školách vůči aktuálním požadavkům na trhu práce.

Klíčová slova

Webové technologie, Tvorba webových stránek, Programování, Kódování

Annotation

VLADIMÍR, M. *Comparison of technologies for teaching web development*. Diploma thesis, University of Hradec Králové, Hradec Králové, 2021.

The thesis monitors the current situation in the field of web development. It describes and compares technologies that should be included in the teaching of web creation. It explains why the technologies are suitable for teaching and gives examples of applications. It compares the ratio of taught technologies in schools to current requirements in the labor market.

Keywords

Web technologies, Webdevelopment, Programming, Coding

Obsah

ÚVOD

Webové technologie dnes neslouží jen pro vytváření webů. Lze pomocí nich vytvořit například mobilní či desktopovou aplikaci[1] a JavaScript původně určený pro zvýšení interakce na webové stránce lze dnes využít i na serveru. Cílem této diplomové práce je vytvoření přehledu aktuálně nejpoužívanějších technologií pro vývoj webových stránek, a z toho doporučení pro jejich zařazení do výuky.

Technologie a nástroje zmíněné v této práci byly vybrány na základě průzkumu pracovních inzerátů aktuálně k roku 2020. Za účelem zjednodušení výběru programovacího jazyka a technologie pro výuku byly dané možnosti seřazeny tak, aby byly vždy podobné alespoň po stránce syntaxe a instalace. Díky tomuto vytřídění se zvyšuje transfer nabytých znalostí z jedné technologie a urychluje se výuka dalších zvolených technologií a nástrojů.

Na začátku této práce jsou aktuálně firmami preferované programovací jazyky. Jelikož se dnes pro zjednodušují a zvýšení efektivity vývoje často používají již připravené a hotové softwarové balíčky, tak je zde jako další zařazena sekce s instalačními nástroji pro jednotlivé jazyky. Z výzkumu vyplývá, že firmy počítají alespoň s povrchní znalostí jednoho frameworku, a to jak po stránce backendové, tak i frontendové, proto je zde zařazena sekce s frameworky.

Nedílnou součástí vývoje dnešních webových aplikací je také optimalizace rychlosti načítání, velikosti webové stránky, optimalizace pro různé prohlížeče a SEO optimalizace. Toho se dosahuje spojováním více načítaných souborů (např. CSS souborů) do jednoho a minifikací kódu v něm obsaženého.[2] Jsou k tomu využívané balíčkovací systémy, jelikož každý je vhodný pro jiný účel, jsou uvedeny a porovnány v samostatné sekci.

---

[1] Pro vytvoření desktopové a mobilní aplikace pomocí HTML, CSS a JavaScriptu lze využít například Electron.js (viz [w-52]). Weby je dnes možné upravit i do takové podoby, aby se alespoň částečně chovali jako nativní desktopové aplikace (viz [w-53]).

[2] s příchodem HTTP/2 se může zdát, že již není tolik potřeba soubory spojovat dohromady, protože se načítají paralelně na základě priorit. s více externími soubory se ale stále zvyšuje počet requestů a v případě pomalejších internetových připojení to stále může zpomalit načtení webu. V případě používání HTTP/2 je vhodné například soubory rozdělit dle požadované doby cachování.

Do práce byla zařazena i sekci s preprocesory, postprocesory pro jazyk CSS. Pro JavaScript byla přidána sekce s kompilátory a rozšířeními. Pro zpřístupnění aplikace z webového prohlížeče je většinou potřeba využít HTTP server, který umožňuje přístup do aplikace na určitém portu. Proto práce obsahuje sekci, ve které jsou tyto HTTP servery porovnány. Práce dále obsahuje porovnání SQL i NoSQL databází a vysvětluje pojem Serverless databáze.

Během vývoje webové aplikace je rovněž potřeba pracovat s různými verzemi kódu (tzv. „verzování“), například aktualizované verze obsahující opravy chyb na stránce a větvit kód podle například úprav či přidaných funkcí. K vysvětlení dané problematiky slouží sekce nazvaná Verzovací systémy. Teoretickou část uzavírá sekce s prostory a možnostmi, kde lze výslednou webovou aplikaci spustit. V této sekci je uvedeno porovnání již zažitých řešení jako jsou webhostingy s moderními a alternativami.

Praktická část je zaměřena na statistické vyhodnocení shromážděných dat. Tato část je dále rozdělena na část zabývající se aktuálně požadovanými technologiemi, zaměřením vývojářů atd. uváděných v pracovních inzerátech a na trhu práce. Druhá část obsahuje statistiky o tom, které z aktuálně požadovaných technologií a nástrojů jsou na školách vyučovány. Tyto dvě části jsou následně porovnány. Ze vzniklých poznatků je vyvozen závěr, který vysvětluje, jak moc se probíraná látka ve škole shoduje s požadavky na uchazeče na pracovním trhu.

1 Jazyky pro vývoj webu

1.1 HTML a XHTML

HTML je momentálně jeden ze dvou značkovacích jazyků sloužící pro tvorbu webových stránek. Aktuální verze je verze 5. Soubory obsahující HTML mají obvykle koncovku „.html“ nebo „.htm“. (viz DUCKETT, Jon. HTML & CSS: design and build websites. Indianapolis, IN: Wiley, [2011]. ISBN 978-1-118-00818-8.).

Druhým značkovacím jazykem je XHTML. XHTML má příponu „.xhtml“ nebo „.xht“ a vyžaduje validní XML zápis. Pokud se do hlavičky souboru uvede správný doctype (viz [w-54]), je možné využít i přípony pro soubory typu HTML uvedené výše. Jakub Vrána [op-1] o XHTML píše *„Ten je o něco přísnější – vyžaduje například uzavírání všech značek a obalování všech hodnot atributů do uvozovek nebo apostrofů.“*

1.2 CSS

Makzan [op-8] popsal CSS takto *„CSS (Cascading Style Sheet) definuje vizuální prezentaci webových stránek. Určuje styly jednotlivých elementů stránky HTML, včetně stylu v různých stavech (např. po přejetí myší nebo klepnutí).“* Soubory obsahující kód CSS mají koncovku „.css“ (viz [op-9], [op-10]).

1.3 JavaScript

JavaScript je programovací jazyk používaný v prohlížeči. Ara Pehlivanian a Don Nguyen [op-3] o JavaScriptu píšou následující *„Přestože začínal jako jednoduchý jazyk, který nacházel uplatnění ve validaci formulářů a drobné manipulaci s obsahem stránky, vyvinul se, a proto v něm dnes můžete vytvářet bohaté klientské aplikace.“*

Kód napsaný v jazyku JavaScript se spouští na straně klienta v prohlížeči. Je také možné spustit JavaScript na straně serveru[3]. Pro spouštění JavaScriptu na straně serveru se používá nástroj Node.js (viz [op-10]) nebo Deno.

JavaScript, který bude spuštěn u klienta je potřeba testovat v různých prohlížečích. Prohlížeče jsou totiž postavené na různých jádrech, a tedy se liší funckionalita. Prohlížeče jako Google Chrome, Edge a Safari jsou postavené na Webkitu, zatímco

[3] Rozdílů při spouštění JavaScriptu v prohlížeči oproti spouštění na straně serveru je několik. Patří sem např. dostupné funkce, kdy na straně serveru je možné pracovat se soubory a spouštět příkazy ale není zde možné pracovat s objekty jako window a document.

například prohlížeč Firefox využívá jako svoje jádro takzvané Gecko. Tento problém na serveru není a je zde potřeba pouze provádět kontroly během aktualizací serveru.

Možnost spuštění JavaScriptu na serveru popularitu tohoto jazyka zvýšila a přináší nové možnosti:

1. Zlepšení výkonu webové aplikace: díky tomu, že Node.js je asynchronní, tak umožňuje zpracovávat více požadavků najednou což zlepšuje rychlost webové aplikace.
2. SSR: aktuálně se díky možnosti spustit JavaScript na straně serveru dostává do popředí přístup, kdy se komponenty, které se mají zobrazit na straně klienta, vykreslí již na straně serveru, a to stejným frameworkem, který je použit na straně klienta. Takto vygenerované komponenty se následně vloží do stránky a pouze se hydratují. Tento postup je více rozebrán v sekci Frameworky a podsekci.

1.4 PHP

PHP je jeden z aktuálně nejrozšířenějších jazyků využívaných na straně serveru. Autorem je Rasmus Lerdorf a první vydání vyšlo v roce 1994 a původně sloužilo pro sledování (trackování) návštěv na jeho osobní stránce. Aktuálně v době psaní této práce (duben 2021) je PHP dostupné již ve verzi 8.0 (viz [w-1]).

Jeho přednosti jsou jednoduchá instalace, téměř žádná vyžadovaná kostra pro soubor PHP, pro základní potřeby výchozí konfigurace a dostupné funkce z knihoven vytvořených ve vyšších programovacích jazycích jako například C++ a Java (viz [w-2]).

1.5 ASP.NET (Core)

ASP.NET je vyvíjen firmou Microsoft a rozšiřuje vývojovou platformu .NET. ASP je zkratka pro Active Server Pages (aktivní serverové stránky) a první verze byla vydána v roce 2002 (viz [op-6]).

ASP.NET obsahuje navíc oproti .NET nástroje a komponenty pro tvorbu webových aplikací. Umožňuje objektově orientovaný přístup k programování webů, šablony mohou obsahovat logiku a události zavěšené na HTML prvky lze propojit s událostmi

objektů v C#. Má vestavěné postupy a funkce pro autentizaci uživatelů a v nových verzích může běžet také na MacOS ale i na Linuxových serverech (viz [w-4]).

1.6 Ostatní

Ostatními, méně používanými jazyky, které jsou momentálně na trhu práce v oblasti vývoje webových stránek žádané jsou například Java, Golang, Python, C# apod.

2 Nástroje pro správu a instalaci balíčků

Vývojáři při práci na svých projektech často vytvářejí kód tak, aby ho nemuseli psát znovu a mohli ho použít na dalších projektech. Takový kód je většinou umístěn v nějaké složce či repozitáři. Současně jsou v této složce mnohdy i informace o balíčku, informace k instalaci a použití, konfigurační soubory a assety, jako jsou ikonky, fonty či obrázky. Celá tato složka se nazývá „balíček" či „knihovna" (viz [w-5]). Takto vytvořený balíček, případně knihovnu může vývojář nahrát do registrů jako je například npmjs.com nebo packagist.org, odkud je mohou používat další vývojáři. Ke stažení, instalaci, aktualizaci a správě balíčků ve svém projektu používají níže zmíněné nástroje.

2.1 JavaScript a CSS

Pro JavaScript i CSS se používají primárně NPM (viz [op-10]) a Yarn. Oba tyto nástroje ukládají soubory do složky node_modules.

2.1.1 NPM

NPM je výchozí, open source, systém pro instalaci balíčků pro Node.js vytvořený firmou NPM, Inc. v roce 2009 (viz [w-6]).

Pro spouštění instalačních příkazů se primárně využívá CLI. Balíčky tento instalační nástroj stahuje defaultně z NPM registrů. Balíčky lze také instalovat z vlastních repozitářů[4], URL nebo lze instalovat lokální balíčky na základě cesty k package.json.

Konfigurační soubor, se kterým NPM pracuje je package.json. Do tohoto souboru se přidávají sekce jako name, dependencies či scripts (viz [w-7]).

Níže je uvedená ukázka NPM příkazů, které se zadávají do konzole.

```
npm install package
npm uninstall package
npm run customCommand
```

Obrázek 1: Ukázka použití NPM příkazů v konzoli

[4] Repozitáře jsou „složky" většinou obsahující balíček s kódem. Tyto složky jsou umístěny na nějakém uložišti např. na GitHubu či GitLabu. Vývojář může použít adresu repozitáře k instalaci uloženého kódu.

2.1.2 Yarn

Yarn je balíčkovací systém vyvinutý firmou Facebook uvedený na trh v roce 2016.

Balíčky stahuje ve výchozím stavu z vlastních registrů a stejně jako NPM umožňuje balíčky stahovat z přímo zadaných adres či instalovat lokální balíčky. Rozdíly oproti NPM jsou například v názvech příkazů a názvu lock souboru. V následující ukázce jsou YARN příkazy uvnitř systémové konzole. Tyto příkazy jsou adekvátní k NPM příkazům uvedeným výše (viz [w-8]).

```
yarn add package
yarn remove package
yarn run customCommand
yarn customCommand
```

Obrázek 2: Ukázka použití Yarn příkazů v konzoli

2.2 PHP

V aplikacích běžících na jazyce PHP se používá Composer, který slouží ke správě balíčků v aplikaci či spouštění příkazů. Pokud programátor nechce využívat Composer, tak mu jako alternativa nezbývá nic jiného než napsat si vlastní skript pro instalaci a kopírování souborů nebo balíčky stahovat ručně. Z hlediska času a komfortu při vývoji ale mohou být tato řešení složitá anebo pomalá.

2.2.1 Composer

Composer je momentálně nejvíce využívaný nástroj pro instalaci PHP balíčků.

```
composer require some/package
composer remove some/package
composer customCommand
```

Obrázek 3: Ukázka použití Composer příkazů v konzoli

Balíčky primárně stahuje z packagist.org. Umožňuje ale také instalovat balíčky z vlastních i privátních registrů.

Během instalačního procesu Composer vytváří i pomocné soubory pro zavádění nainstalovaných závislostí do aplikace. Tyto soubory obsahují například pole s cestami k souborům, ve kterých jsou umístěny požadované třídy. Tyto pomocné soubory,

konkrétně např. autoload.php je využíván v PHP aplikacích a frameworcích při vytváření tzv. DI kontejnerů apod. (viz [w-9]).

2.3 ASP.NET (Core)

2.3.1 NuGet

NuGet je oficiální správce a instalátor balíčků pro „.NET“. Stejně jako výše zmíněné nástroje stahuje balíčky primárně z vlastních registrů.

Registry mají vlastní web nuget.org, kde je možné balíčky vyhledávat. Instalace zvoleného balíčku opět probíhá pomocí CLI (viz [w-10]).

3 Frameworky

Pojmem framework se nemyslí jen jeden jediný framework (např. .NET framework). Frameworky jsou obecně předpřipravený a většinou i mnoha vývojáři otestovaný kód, který vývojáři stačí nainstalovat či vložit do stránky a začít používat.

Vývojáři tím šetří čas a firmy peníze. O vývoj frameworků se buďto starají jednotlivci, komunity nebo firmy, které tento kód vytváří a prodávají pod placenou licencí. Pro rozsáhlejší a více používané frameworky existují i školení a certifikace (viz [w-11]).

3.1 CSS

V poslední době se v oblasti webových technologií objevuje tzn. atomický přístup k CSS. Znamená to, že se využívá spíše utilitový přístup[5] namísto komponentového[6]. V této sekci je porovnání a odůvodnění utility first frameworku Tailwind a component first frameworku Bootstrap. Zároveň jsou zde uvedeny i další alternativy.

3.1.1 Tailwind

Tailwind je open sourcovaný, utility first, CSS framework založený Adamem Warhanem a Jonathanem Reininkem (viz [w-12]).

Tento framework namísto přístupu předpřipravených a komponent s již předdefinovaným stylem jako jsou tlačítka, formuláře, navigace apod. poskytuje utility pomocí nichž se dají dané komponenty vytvořit.

Výhodou tohoto přístupu je například to, že nemusíme neustále přepisovat styly u komponent, které chceme visuálně upravit oproti základní podobě. Pokud má tedy například komponenta zaoblení rámečku border-radius: 2px a my budeme chtít border-radius: 4px, pak je potřeba tento styl přepsat. V případě Tailwindu zde žádná předpřipravená komponenta není, a proto ji musíme vytvořit znovu s například již poskytnutými utilitami.

---

[5] V CSS kódu, který využívá utilitový přístup má každá CSS třída pouze jednu vlastnost. Například .text-left zarovná text vlevo a třída .font-bold převede text elementu s touto třídou na tučný. Utilitové třídy mohou mít ještě prefixy pro media-queries např. md:text-left. Tím vývojář říká, že tato třída ovlivní element až od středního breakpointu, který může být např. od 640 px (mobilní telefon).

[6] U komponentového přístupu mají třídy na rozdíl od utilitového přístupu více vlastností. Například třída .btn může u elementu s touto třídou měnit velikost písma, pozadí či stín elementu. Styl třídy .btn by se v utilitovém přístupu zapsal pomocí několika tříd na jednom elementu

Další výhoda oproti například níže zmíněnému Bootstrapu a jemu podobným komponentově založeným frameworkům je ta, že je nastavitelný pomocí jednoho souboru tailwind.config.js. Díky tomu není konfigurace rozmístěna různě po aplikaci a zlepšuje se tak přehlednost (viz [w-13]).

```
<button class="
    bg-blue-500 hover:bg-blue-600
    text-white font-bold md:font-lg
    py-2 px-4
    rounded
">
    Button
</button>
```

Obrázek 4: Ukázka použití Tailwind frameworku a utility first přístupu

3.1.2 Bootstrap

Bootstrap je asi jeden z nejznámějších zástupců component first CSS frameworků. Původním autorem byli vývojáři Twitteru. Nyní ho spravuje menší skupina vývojářů na GitHubu (viz [w-14]).

Od poslední majoritní verze 4, která vyšla v roce 2015, využívá tzv. Reboot.css, což je vylepšená obdoba Normalize.css. Rozdílů je zde několik, přičemž těmi zásadními jsou, že výchozí jednotkou pro font je rem[7] a směr vnějších okrajů směřuje dolů.

I když Bootstrap poskytuje připravené komponenty jako jsou tlačítka, tabulky, navigace a díky tomu umožňuje rychlý a pohodlný vývoj webu pouze pomocí přidávání tříd, tak se tato jeho výhoda občas stává nevýhodou.

Jak již bylo zmíněno v sekci s frameworkem Tailwind, tak mnohdy je potřeba upravit vzhled komponenty. Z pohledu CSS můžeme např. potřebovat na jedné stránce stejné tlačítko, ale s větším písmem než na jiné. Můžeme také potřebovat jinou barvu či více zaoblené rohy. Tím nám vzniká kód navíc, který může s růstem projektu narůstat a komplikovat se. Dalším problémem je, že Bootstrap a jemu podobné frameworky pro některé komponenty využívá JavaScript, který je závislý např. na knihovně jQuery. Tato

[7] Rem je jednotka, jejíž velikost vychází z velikosti písma, která je nastavená pro dokument. Další jednotky jsou em, vw, vh, % a px (viz [w-55]).

závislost způsobuje problém v momentě, kdy chtějí vývojáři na webu využít jiný framework než jQuery (viz kapitola 3.2.1). Pokud by chtěli využít i na JavaScriptu závislé komponenty, znamená to pro ně přidat závislost na knihovně jQuery, čímž se jim zvýší počet i velikost externích závislostí a zároveň se jim zhoršuje udržitelnost aplikace.

```
<button class="btn btn-primary">Button</button>
```

Obrázek 5: Ukázka použití Bootstrap frameworku a komponentového přístupu

3.1.3 Ostatní

Mezi další frameworky, které se vyskytují na celosvětové scéně, ale nejsou tak populární, patří například Foundation (viz get.foundation) a Tachynos (viz tachyons.io).

Zajímavým frameworkem, který generuje CSS dynamicky je také Atomic.css (viz acss.io). Tento framework generuje CSS dynamicky na základě přidaných tříd v HTML dokumentu. Díky tomu se snižuje výskyt nepoužitého CSS na stránce a programátor má větší flexibilitu. Nemusí kvůli jedné třídě navíc upravovat generátory apod.

3.2 JavaScript

Díky pokroku ve vývoji JavaScriptu a snížení počtu používání zastaralých prohlížečů se začali na trhu vedle známé knihovny jQuery čím dál častěji objevovat progresivní frameworky založené na principu překreslování šablon a nepřímou manipulací s DOMem[8].

3.2.1 jQuery

jQuery je jedna z nejznámějších JavaScriptových knihoven. Tato knihovna je vyvíjená od roku 2005 Johnem Resigem. jQuery byla na začátku pouze jakýmsi setem nástrojů, které John používal pro svou každodenní práci (viz [w-15]).

Postupně se vyvinula ve framework, který zjednodušoval vývojářům práci tím, že zajišťoval kompatibilitu vytvořeného kódu napříč různými prohlížeči. Díky tomu se

[8] Nepřímá manipulace s DOMem znamená, že vývojáři využívají například virtuální DOM (VDOM), který po sestavení a aktualizaci následně sesynchronizují s DOMem aktuální stránky (viz [w-62], [w-64]).

psaní JavaScriptu zjednodušilo, zrychlilo, a i komplexní kód mohl být napsán méně pokročilými programátory (viz [w-16]).

I v roce 2020 se jedná o stále udržovanou knihovnu, která je hojně využívána a aktuálně je dostupná ve verzi 3.6.0. Zároveň se již také připravuje verze 4.0.

Díky tomu, že funkcionalita JavaScriptu je v dnešních webových prohlížečích celkem jednotná, tak se vývojáři snaží od jQuery upouštět a spíše využívají reaktivní frameworky jako je níže zmíněný Vue.js (viz kapitola 3.2.2). Není také vhodné jQuery s těmito frameworky kombinovat, protože by úprava DOMu pomocí jQuery mohla způsobit rozdíl ve stavu aplikace oproti předpokládanému stavu uvnitř VDOMu[9] a datového objektu.

V případě, kdy se vývojáři rozhodnou pro volbu jednoho z frameworků fungující na principu zmíněném v sekci Vue.js, se zužuje možnost volby CSS frameworků. Zejména se vyřazují ty, které jsou jakkoliv závislé na JavaScriptu a nemají alternativu pro daný framework. Přechází se tedy spíše k utility first CSS frameworkům, který tímto neduhem netrpí, či kompletně vlastním stylům.

```
<script src="https://code.jquery.com/jquery-3.5.1.slim.min.js"></script>
<script>
    $(() => $('.btn').click(() => alert('Ahoj!')));
</script>
<button class="btn">Klikni</button>
```

Obrázek 6: Ukázka přidání posluchače na kliknutí v jQuery

3.2.2 Vue.Js

Vue.js je jedním ze zástupců progresivních frameworků. Byl vytvořen Evanem You a na trh uveden v roce 2014.

[9] Každá stránka má po načtení vytvořený strom všech prvků na stránce (DOM nodes). Do tohoto stromu spadají elementy, text i komentáře na stránce. Aktualizace jednotlivých prvků může být výkonově náročná. Proto vývojáři vytváří vlastní „strom prvků“, který je tvořen např. „objekty“ popisující dané prvky. Změny prováděné v tomto virtuálním domu nejsou tak výkonově náročné. V momentě, kdy programátor potřebuje namísto přidat prvek do košíku, zvýšit cenu objednávky a upravit informace k dopravě, tak se všechny tyto změny nejdříve propíšou do virtuálního domu a následně se celý VDOM sesynchronizuje s DOMem stránky (propíše se do stránky) a uživatel uvidí aktualizované informace (viz [w-62], [w-63], [w-65]).

Vue.js se oproti například výše zmíněné jQuery liší jak syntaxí, tak funkcionalitu.

```
<div id="app">
    <button class="btn" @click="click">Klikni</button>
</div>
<script src="https://cdn.jsdelivr.net/npm/vue/dist/vue.js"></script>
<script>
    new Vue({
        el: '#app',
        methods: {
            click: () => alert('Ahoj!')
        }
    });
</script>
```

Obrázek 7: Ukázka adekvátního kódu ve Vue.js k jQuery kódu výše

Vue.js využívá instance, které mají například datový objekt, šablonu, hooky[10], metody pro manipulaci s daty apod. Na rozdíl od jQuery také programátor využívající Vue.js nepracuje s DOMem přímo, ale pouze mění například data v datovém objektu na základě čehož Vue.js překresluje šablonu (viz [w-17]).

[10] V průběhu zpracovávání Vue.js instance se vykonávají jednotlivé události v životním cyklu instance. Patří sem například událost po vytvoření instance, vložení do stránky, překreslení na základě dat atd. Vývojář může na jednotlivé událost napojit funkce, které se spustí v momentě, kdy komponenta zrovna prochází danou částí životního cyklu. Může tak například zobrazit informaci o tom, že produkt byl vložen do košíku, hned jak se uloží a zobrazí na stránce (viz [w-60]).

```
<div id="app">Číslo: {{ number }}</div>
<script src="https://cdn.jsdelivr.net/npm/vue/dist/vue.js"></script>
<script>
    const app = new Vue({
        el: '#app',
        data() {
            return {
                number: 1
            }
        },
        watch: {
            number(val) {
                alert(`Číslo se změnilo na: ${val}.`);
            }
        },
        mounted() {
            alert('Aplikace je připravena.');
        }
    });
    app.number++;
</script>
```

Obrázek 8: Ukázka překreslení šablony při změně čísla v datovém objektu ve Vue.js

Překreslení probíhá tak, že pokud se změní data v datovém objektu, spustí se již zmíněné překreslení šablony. Změny se nejdříve propíší do VDOMu, následně se VDOM porovná s DOMem a výsledné rozdíly se vykreslí podle nových dat na stránce. Tedy pokud se má například přidat třída, tak se přidá. Pokud se má změnit obsah uvnitř elementu podle nového obsahu v datovém objektu instance, tak se změní. Díky tomu je vykreslování velmi rychlé.

Není také dobré kombinovat typy frameworku jako je Vue.js s jQuery, protože pokud například odebereme třídu nebo potomka v elementu, který byl vykreslený pomocí Vue.js, tak Vue.js o této změně neví a vnitřní stav instance je oproti aktuálnímu stavu na stránce jiný. Tedy při dalším vykreslení obsah nemusí odpovídat.

K práci se sdíleným stavem napříč aplikací se pak dále využívá knihovna Vuex. Tato knihovna je určená pro centralizovanou správu určitého stavu uvnitř aplikace. Typickým příkladem může být nákupní košík. V momentě přidání či odebrání položky se musí přepočítat cena za zboží. Tato cena se následně musí propsat na více míst na

webu. Spravovat tento stav ručně nebo přes posluchače by bylo náročné a časem, jak se aplikace vyvíjí, také nepřehledné a mohlo by to vést k refaktoringu. Vuex zjednodušuje řešení toho problému. Změny uvnitř Vuex instance se provádí přes settery a data se získávají přes gettery. Při změně stavu uvnitř instance Vuex se překreslí všechny instance, které jsou na tuto instanci napojené. Pokud tedy použijeme Vuex pro nákupní košík a obsah košíku se změní, tak aktualizovaný obsah propíše na všechny instance. Alternativou pro Vuex je například Flux (viz facebook.github.io/flux) či Redux (viz redux.js.org), ze kterých se autoři Vuex inspirovali (viz [w-18]).

3.2.3 AMP

AMP je komponentovým framework od Googlu. Tento framework si klade za cíl zrychlit načítání webů. Na rozdíl od zjednodušování psaní JavaScriptu a komponent, jako oba výše zmíněné frameworky, upřednostňuje striktnost a spíše namísto rozšiřování možností pro psaní aplikace, tvorbu aplikace dělá více striktní a zároveň možnosti omezuje (viz [w-19]).

Jak již bylo zmíněno výše, zrychlení vykreslení webů se snaží dosáhnout několika způsoby. V první řadě zavádí striktní pravidla pro to, co může obsahovat samotné HTML. Tedy například nepovoluje klasické obrázky. Je to z toho důvodu, že jednak načtení obrázku bez lazyloadingu zpomaluje načtení webu a zadruhé, pokud vývojář nevytvoří pro načítaný obrázek prostor, tak po načtení obrázku obsah poskočí a spustí se reflow stránky, což opět zpomaluje celkové načtení webu (viz [op-2]).

Další ze striktních pravidel je omezení velikosti CSS, které může mít velikost v době psaní této závěrečné práce maximálně 50 kB a musí být vloženo v jednom style tagu. Také se v CSS kódu nesmí vyskytovat klíčové slovo „!important“ (viz [op-7]).

AMP také omezuje možnost psaní vlastního JavaScriptu. Možnost napsat vlastní skript je umožněna pouze za pomocí tagu amp-script Vlastní napsaný JavaScript je opět omezen striktními pravidly jako je doba provedení a velikost.

JavaScript je na webu omezen schválně. AMP namísto možnosti vlastního JavaScriptu poskytuje již zmíněné předpřipravené komponenty. Tyto komponenty sice nemají takovou funkcionalitu jako běžně dostupná řešení, ale zato jsou odladěná a mají malou

velikost. V případě, že chce vývojář takovou komponentu použít, musí ji vložit do hlavičky stránky a následně nastavit podle dokumentace.

Jelikož se při programování chybuje a programátoři občas na něco zapomenou, tak AMP také přináší validátor. Tento validátor vypisuje chyby v momentě, kdy na stránce je například použit img tag namísto amp-img tagu a také, když programátor načte komponentu, ale následně ji na dané stránce nevyužije.

Pokud je AMP stránka na daném webu validní, tak ji Google přednačte a následně načítá ze svého CDN[11] a tím snižuje dobu načtení stránky. Dotaz pro načtení obsahu webu pak nesměřuje na server webu ale na servery Google. Tato část AMP se nazývá AMP Cache.

Kromě webu je AMP možné použít i pro tvorbu responzivních e-mailů či optimalizovaných reklam. Bezkonkurenční výhodou AMP e-mailů je možnost používat moderní CSS vlastnosti bez nutnosti tabulkového layoutu. V AMP e-mailech lze také využít JavaScriptové komponenty, díky kterým může vývojář přidat do e-mailu interaktivní prvky (např. carousel, záložky či Ajaxově získávaný obsah). Bohužel i přes propagaci Googlem stále nejsou AMP komponenty podporovány ve všech klientech a na Českém trhu je podíl uživatelů, kterým se AMP e-maily zobrazí, v porovnání se zobrazením běžných e-mailů, velmi malý (viz statistiky ze slevomat.cz [w-56]).

[11] CDN je zkratka pro Content Delivery Network, což je síť serverů, na kterých se cachuje obsah, který se následně posílá uživatelům do prohlížeče. Pokud je například web v Česku a má se jeho obsah jako jsou např. obrázky rychle zobrazit v Japonsku, tak aby se na data nečekalo a nenačítala se až z Česka, tak se využije se server z CDN sítě, který je například v Japonsku (viz [w-61]).

```
<!-- Načtení závislosti amo-carousel -->
<script
    async
    custom-element="amp-carousel"
    src="https://cdn.ampproject.org/v0/amp-carousel-0.1.js"
></script>
<!-- Použití -->
<amp-carousel height="400" layout="fixed-height" type="carousel">
    <amp-img src="/image1.jpg" width="400" height="300" alt=""></amp-img>
    <amp-img src="/image2.jpg" width="400" height="300" alt=""></amp-img>
</amp-carousel>
```

Obrázek 9: Ukázka práce s AMP komponentami

3.2.4 Ostatní

Dalšími aktuálně rozšířenými alternativami výše zmíněných frameworků jsou React, Angular a Svelte. React je výtvorem z dílny Facebooku, Angular byl tvořen Googlem a Svelte založil Rich Harris. React, Angular i Svelte jsou progresivní frameworky s reaktivním datovým objektem. Liší se primárně funkcionalitou, výkonem a syntaxí. Svelte v rámci své vnitřní funkcionality z této trojice poněkud vystupuje. Jako jediné nevyužívá VDOM ale změny okamžitě propisuje do DOMu.

3.3 PHP

Paralelně s vývojem PHP se vyvíjí frameworky postavené na tomto jazyce. Frameworky začínají postupně implementovat striktní datové typy a integrovat nové PHP funkce.

3.3.1 Symfony

Symfony je PHP framework, který je vyvíjen a sponzorován primárně firmou Sensio Labs. První verze byla vydána v roce 2005 a aktuálně primární verze má číslo 5.2.

Verze 4.x a aktuálně vyvíjená řada 5.x se liší především tím, že oproti předchozím verzím neobsahují nepotřebné komponenty, které programátor nemusí využít. Nejmenší instalační balíček toho frameworku se jmenuje symfony/skeleton a obsahuje pouze jádro a komponenty nutné pro http aplikace, který se instaluje přes již zmíněný

nástroj Composer. Základní kostru aplikace lze využít například pro tvorbu API.[12] V momentě, kdy vývojář potřebuje šablonovací systém a další komponenty, může si je nainstalovat. Tento krok razantně zmenšil, zjednodušil a zpřehlednil tvorbu aplikací v tomto frameworku (viz [w-20]).

Ve verzi 4.0 byl přidán balíček nazvaný symfony/flex, který automatizuje procesy jako kopírování konfiguračních souborů, přidání závislostí do autoload souboru atd. Tím odstraňuje řadu úkonů, které musel programátor provádět ručně, aby balíček správně nainstaloval. Zároveň ale tento balíček do aplikace přidává další logiku a dělá ji tak více komplexní. V případě, že programátor chce zautomatizovat instalaci svého balíčku, musí vytvořit požadavek na spojení změn (tzv. merge request) na GitHubu do repozitáře obsahujícího instalační skripty od komunity. V merge requestu musí být obsažena základní struktura dle návodu a instalační příkazy (viz [w-21]).

V porovnáním s Nette má tento framework větší celosvětovou komunitu, více vývojářů, sponzorů, více vlastních školení a je celkově více využíván. Neobsahuje však českou dokumentaci.

```
<span>{{ text }}</span>
{% if text %}
    Délka textu: {{ text|length }}
{% endif %}
```

Obrázek 10: Ukázka syntaxe šablonovacího systému Twig

3.3.2 Nette

Nette je PHP framework primárně vyvíjený Davidem Grudlem a sponzorovaný různými firmami a komunitou. První verze byla zveřejněna v roce 2008 po 4 letech vývoje a aktuální verze je 3.0. Nette však oproti výše zmíněnému Symfony frameworku nemá monorepozitář, takže verze některých balíčků toho frameworku se mohou lišit (viz [w-22]).

[12] API je zkratka pro Application Programming Interfaces. API je připravená aplikace, která slouží např. pro komunikaci vývojáře s interní databází nebo mezi e-shopem a skladem pomocí dotazů. Pokud například e-shop potřebuje zobrazit počet zboží na vzdáleném skladu, aplikace pošle dotaz na API skladu a ten vrátí aktuální počet naskladněných produktů. API je také většinou přizpůsobeno na vyšší počet dotazů než běžná stránka (viz [op-11], [op-12]).

Nejmenší a také základní instalační balíček toho frameworku se jmenuje nette/web-project. Stejně tak jako u Symfony se tento balíček instaluje před Composer. Balíček již v základu obsahuje vše pro vytvoření webové aplikace včetně závislosti na šablonovacím systému. Oproti Symfony má však celkově méně závislosti a také nemá žádný komplexní nástroj pro instalaci a kopírování konfiguračních souborů.

V porovnání se Symfony má Nette méně vlastních balíčků, menší komunitu a je využíváno především v České republice. Avšak díky komunitě a práci vývojářů je možné používat velké množství komponent a balíčků z jiných frameworků jako rozšíření do frameworku (viz [w-23]).

Z technologického hlediska jsou si oba frameworky více méně podobné a fungují na podobných principech. Rozdíly jsou například v použitém šablonovacím systému, kdy Nette využívá Latte a Symfony používá Twig. Symfony oproti Nette používá architekturu MVC[13], zatímco Nette je postavené na architektuře MVP[14]. Symfony zároveň následuje některé standard PSR[15]. Díky tomu je možné v Symfony použít některé balíčky, které v Nette nikoliv. Příkladem může být využívání React PHP, který s Nette není kompatibilní, zatímco se Symfony ano.

```
<span>{$text}</span>
{if $text}
    Délka textu: {$text|length}
{/if}
```

Obrázek 11: Ukázka syntaxe šablonovacího systému Latte

3.3.3 Ostatní

Mezi další známé a populární frameworky se řadí například Laravel, Codeigniter, Yii 2, Cake PHP. Tyto frameworky fungují na obdobných principech. Rozdíl lze nalézt například v adresářové struktuře, počtu dostupných balíčků a vnitřní funkcionalitě.

---

[13] Model-View-Controller je způsob psaní aplikace, kdy se odděluje datová vrstva od prezentační. Komunikaci mezi těmito vrstvami řídí Controller. Činnost controlleru je například posílat data získaná z datové vsrtvy (např. databáze) do šablon, přesměrovávat mezi stránkami atd. (viz symfony.com/doc/current/create_framework/introduction.html).

[14] Nette využívá koncept Mode-View-Presenter. V tomto případě ještě view navíc zpracovává akce od uživatele (viz doc.nette.org/cs/3.1/presenters).

[15] PHP Specification Request (PSR) (viz [w-24]).

4 CSS Preprocesory a postprocesory

Webový vývoj se dnes ubírá směrem rozdělování aplikace na jednotlivé a znovupoužitelné komponenty. U CSS tomu není jinak. Ke zjednodušení tohoto způsobu vývoje se používají preprocesory a postprocesory.

Preprocesory mají vlastní syntaxi a pomocí nich se CSS teprve generuje. Postprocesory se používají po vygenerování CSS a slouží primárně k přidání prefixů do CSS souboru pro starší prohlížeče.

Tyto nástroje umožňují sdílet například proměnné, funkce a části CSS kódu napříč aplikací. Celkově zjednodušují vývoj, zmenšují výskyt duplicitního kódu a snižují vznik chyb.

4.1 SASS

SASS je preprocesor pro CSS který v roce 2006 vytvořil Ampton Catlin, Natalie Weizenbaum, Chris Eppstein a Jina Anne.

Tento preprocesor poskytuje kromě vylepšené syntaxe zápisu jazyka CSS proměnné, mixiny, vlastní funkce, placeholdery a výpočty během kompilace kódu.

SASS poskytuje 2 typy syntaxe. Se závorkami (SCSS) a styl SASS, který namísto závorek používá zanořování a odsazování. SCSS je ze dvou zde představených CSS (pre)procesorů používaný častěji a také poskytuje některé funkce které v SASS z důvodu absence závorek nejsou dostupné. Vylepšená syntaxe spočívá v tom, že je možné selektory libovolně zanořovat do sebe a navazovat na předešlé selektory a dědit vlastnosti již existujících selektorů.

V CSS je sice možnost vytvářet proměnné ale ty nejsou kompatibilní ve všech prohlížečích. I v roce 2020 jde stále spíše o experimentální proces a vývojáři zůstávají u SCSS proměnných. Tyto proměnné je možné vkládat v kódu na libovolná místa a následně jsou nahrazeny hodnotou proměnné (viz [w-25]).

```scss
$bold: 700;

.text {
    &-bold {
        font-weight: $bold
    }
}
```

Obrázek 12: Ukázka SCSS syntaxe preprocesoru SASS

Rozdíl mezi mixiny a funkcemi v jazyce SASS je, že funkce jsou primárně používány pro vracení nějaké hodnoty v jakémkoliv podporovaném typu, zatímco mixiny jsou používány pro generování výstupu. Typickým příkladem pro použití funkcí je například přepočet jednotek z px na rem. Mixinami zase můžeme obalit část CSS kódu a tento kód ovlivnit uvnitř bloku a vypsat jej na základě podmínky třeba pro různý media query.

```scss
@function decreaseWidth($width) {
    @return $width - 1;
}
@mixin breakpoint($width, $type: "min-width") {
    @if $width == 0 {
        @content
    } @else {
        @if $type == "max-width" {
            $width: decreaseWidth($width)
        }
        @media (#{$type}: #{$width}px) {
            @content
        }
    }
}
.block {
    width: 400px;
    @include breakpoint(500, max-width) {
        width: 500px;
    }
}
```

Obrázek 13: Ukázka použití mixinu a proměnné v preprocesoru SASS

Zmíněné placeholdery u SCSS fungují jako znovupoužitelný kus kódu. Pokud placeholder umístíme do kódu, tak po kompilaci bude na místě, kde byl použit placeholder CSS daného placeholderu.

4.2 PostCSS

PostCSS je CSS postprocesor vytvořený Andreyem Sitnikem.

Tento nástroj funguje obdobně jako JavaScriptové kompilátory. Přetváří CSS napsané pro moderní prohlížeče do takové podoby, aby bylo funkční v prohlížečích, které moderní funkce nemají. Díky tomu opět snižuje vznik chyb, zjednodušuje vývoj a odebírá nutnost si pro každou CSS vlastnost pamatovat prefixy, a jak má být daná vlastnost zapsána pro jiné prohlížeče (viz [w-26]).

Oproti SCSS ale neobsahuje rozšíření jako proměnné, mixiny, funkce apod. Nevyžaduje ani speciální koncovku „sass“ nebo „scss“. Tento nástroj pouze upraví již vygenerované CSS. Samotný SASS preprocesor tento nástroj používá, konkrétně nástroj autoprefixer, který se stará o prefixování CSS dle konfigurace.

Pokud se podíváme na postup vývoje preprocesorů ať už zmíněného sassu výše nebo ostatních alternativ, tak můžeme považovat za výhodu i to, že do aplikace nezanáší nutnost vlastní koncovky ani speciální syntaxi. Může tak být jednoduše odebrán. To je vhodné právě v momentě, kdyby tento nástroj přestal být vyvíjen nebo ho již vývojář nepotřeboval. Totéž však nejde říct o SASS a jiných preprocesorech, kde by vývojáři museli převádět syntaxi na čisté CSS nebo do syntaxe jiného preprocesoru.

4.3 Ostatní

Alternativy pro výše zmíněné nástroje jsou Less a Stylus. Oba nástroje jsou preprocesory fungující na podobném principu. Oproti výše zmíněným se liší syntaxí, koncovkou souborů a množstvím dostupných funkcí.

5 Rozšíření a kompilátory jazyka JavaScript

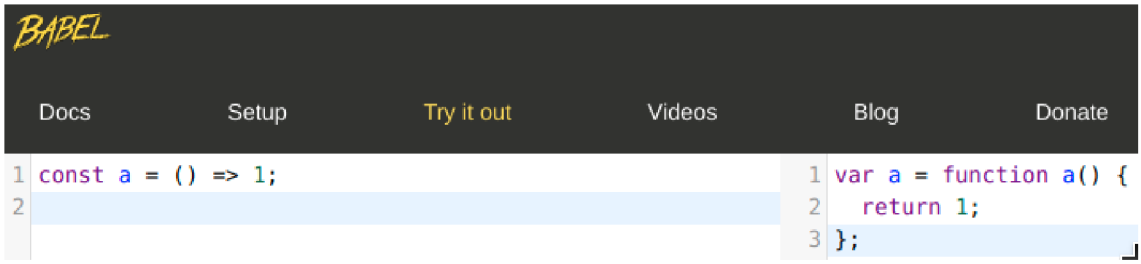
Rozšíření a kompilátory jazyka JavaScript slouží k usnadnění, zjednodušení a zefektivnění vývoje webové aplikace. Kompilátory zjednodušují a dělají vývoj více efektivním díky tomu, že odstraňují úkony, které by musel vývojář dělat ručně. Rozšíření pro tento jazyk naopak dělají kód více striktním a minimalizují vznik chyb.

Rozšíření i kompilátory je možné používat samostatně nebo v průběhu buildu aplikace pomocí balíčkovacího systému.

5.1 Babel

Babel je kompilátor pro JavaScript. Autorem je Sebastian McKenzie.

Tento nástroj umožňuje převést JavaScriptový kód napsaný pro nejnovější prohlížeče, do verze funkční v prohlížečích, které tyto nemají, nepodporují využitou funkcionalitu. Zároveň zachovává funkcionalitu a odebírá nepotřebný kód (viz [w-27]).



Obrázek 14: Ukázka výsledku převedeného kódu Babelem

5.2 Typescript

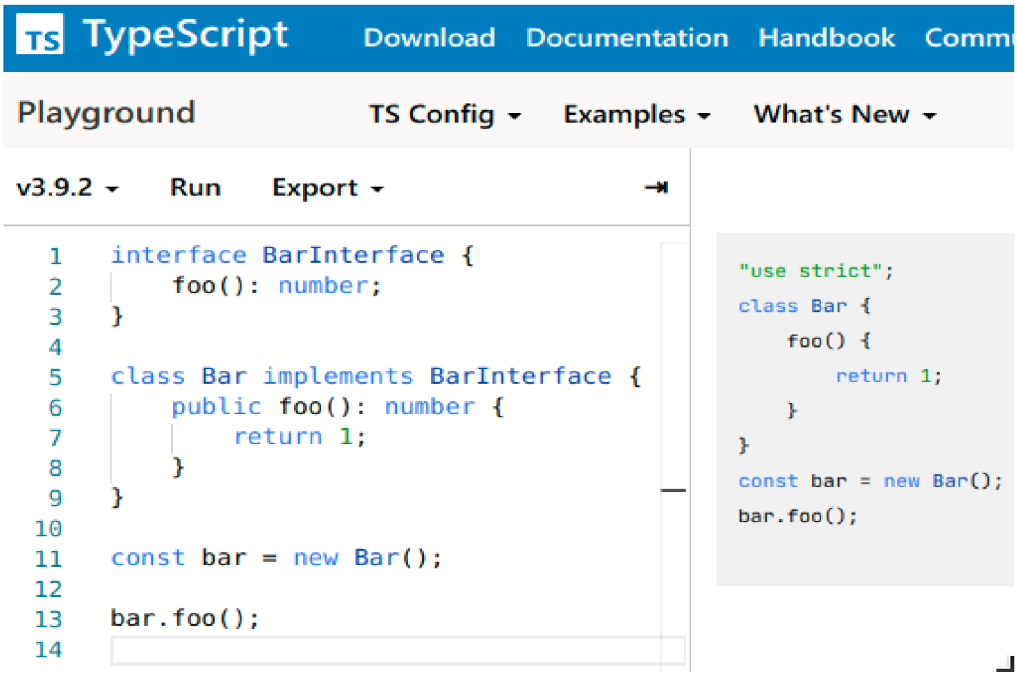
Typescript je nástroj od Microsoftu, který rozšiřuje JavaScript o různé datové typy, struktury a funkcionalitu, které v JavaScriptu aktuálně ještě nejsou zabudované.

```typescript
interface BarInterface {
    foo(): number;
}

class Bar implements BarInterface {
    public foo(): number {
        return 1;
    }
}

const bar = new Bar();
bar.foo();
```

Obrázek 15: Ukázka Typescript syntaxe

Stejně tak jako výše zmíněný Babel výsledný kód převádí do čistého JavaScriptu a opět umožňuje kód vygenerovat pro starší prohlížeče na základě konfigurace.

Rozdílem oproti Babelu je syntaxe, která může být stejná jako je v Babelu, ale v momentě, kdy se přidají speciální datové typy, tak je odlišná. Kód využívající rozšíření Typescript musí být souboru s koncovkou „.ts“ a definiční soubory s koncovkou „.d.ts“ (viz [w-28]).



Obrázek 16: Ukázka výsledku převedeného kódu Typescriptem

6 Balíčkovací nástroje

Balíčkovací systémy (bundlery) slouží pro zpracování souborů různých typů. Umí například spojit více CSS nebo JavaScriptových souborů dohromady a udělat z nich jeden soubor. Tyto systémy rovněž umožňují tvorbu vlastních pluginů. Díky tomu je možné vytvořit plugin například pro výše zmíněný SASS nebo Typescript a před vytvořením výsledných souborů JavaScript či CSS zkompilovat a zmenšit pomocí minifikátorů.[16]

Kromě stejných typů umí spojit i různé typy souborů. Standardem je dnes spojování obrázků, CSS, fontů i JavaScriptu do jednoho souboru. Tento soubor má většinou příponu „.js“. Po načtení webu je spuštěn skript, který vloží do dokumentu dané závislosti. Tento přístup je vhodný například pro vytváření knihoven a balíčků, kdy potřebujeme dodat všechny závislosti v jednom souboru anebo chceme zmenšit počet requestů na stránce.

Většina těchto systémů také podporuje tzn. Tree-Shaking, což je odstraňování nepoužitého kódu a nepotřebných závislostí (viz [w-29]).

Protože se dnes klade důraz na co nejmenší velikost stránky, tak tyto nástroje také podporují takzvaný Code-Splitting. Tato technologie umožňuje rozdělit kód na co nejmenší části a načítat dané části kódu až v momentě, kdy jsou opravdu potřeba. Načítání proběhne například pomocí Ajaxu (viz [w-30]).

Jednou z neopomenutelných výhod balíčkovacích nástrojů je i ta, že v případě, kdy vývojář vytváří například SDK pro jeho nástroj, tak většinou potřebuje různé formáty vygenerovaného JavaScriptu. Pro Node.js potřebuje ES6 nebo CJS moduly. Pro prohlížeč pak AMD, UMD a ES6 moduly upravené pro prohlížeč. Díky balíčkovacím systémům mu stačí přidat konfiguraci pro daný formát a nemusí ručně vytvářet inicializační struktury pro dané formáty (viz [w-31]).

[16] Minifikátory zmenšují kód například pomocí odstraněním mezer, zkrácením názvů proměnných a funkcí apod.

6.1 Webpack

Webpack je jedním z nejznámějších a nejpoužívanějších balíčkovacích systémů. Autorem je Tobias Koppers a první vydání bylo vydáno v roce 2012.

Oproti konkurenci má Webpack více aktivních vývojářů, rozsáhlou komunitu a velké množství pluginů.

Webpack vytváří .js soubory s veškerými moduly a jejich závislostmi. Každý modul, který má Webpack do výsledného .js souboru přidat, je obalen inicializační funkcí. Je to z důvodu snížení rizika například kolize názvů proměnných či funkcí. Tento postup zároveň zvětšuje velikost výsledného souboru. Webpack v základu vkládá CSS do přímo do JavaScriptu. Následně se CSS do stránky vkládá pomocí JavaScriptové funkce. Output Webpacku lze ale rozdělit pomocí pluginů. Pokud vývojář potřebuje separátní CSS soubor stačí, když si nainstaluje plugin, který umí oddělit CSS do .css souboru (viz webpack.js.org/plugins/mini-css-extract-plugin).

Konfigurace Webpacku se vkládá do souboru do webpack.config.js. z tohoto souboru je exportován objekt nebo funkce s veškerou konfigurací (viz [w-32]).

```
const path = require('path');

module.exports = {
    entry: './src/index.js',
    output: {
        filename: 'main.js',
        path: path.resolve(__dirname, 'dist'),
    },
};
```

Obrázek 17: Ukázka Webpack konfigurace

Při volání příkazu pro spuštění Webpacku se pak zadá cesta ke konfiguračnímu souboru následovně.

Od verze 4.x je v plánu, že Webpack bude fungovat i bez jakékoliv konfigurace obdobně jako níže zmíněný Parcel.

6.2 Rollup

Rollup je vedle Webpacku dalším velmi rozšířeným balíčkovacím nástrojem. Autorem je Rich Harris.

Stejně jako Webpack umí Rollup spojovat dohromady různé typy souborů do .js souboru. Podobně jako Webpack má možnost pluginů. Rozdílem v build procesu je, že jednotlivé moduly nejsou obalovány inicializační funkcí. Výsledný soubor je tak menší než u Webpacku. Díky tomu je Rollup vhodnější pro tvorbu knihoven a závislostí, které musí být velikostí co nejmenší (viz [w-33]).

V porovnání s Webpackem nemá Rollup tak velkou komunitu ani počet pluginů.

```
export default {
    input: 'src/index.js',
    output: {
        file: 'dist/main.js',
        format: 'umd'
    }
};
```

Obrázek 18: Ukázka Rollup konfigurace

6.3 Parcel

Parcel byl vytvořen Devonem Govettem. Je to balíčkovací systém, jehož myšlenkou je zpracovávat assety bez nutnosti cokoliv konfigurovat (viz [w-34]).

Potřebné balíčky pro kompilaci assetů instaluje za běhu. Tedy pokud potřebujeme zpracovat soubor jazyka SASS a nemáme nainstalovaný kompilátor pro SASS, tak si ho Parcel nejdříve nainstaluje a následně provede kompilaci.

Nevýhodou Parcelu je, že se obtížně integruje do PHP frameworků, které využívají šablonovací systémy jako Latte nebo Twig. V cestách k souborům, ať už to jsou obrázky nebo CSS či JavaScript soubory se mnohdy vyskytují proměnné a Parcel s tím neumí pracovat. Proto se Parcel hodí především na jednoduché projekty a převážně statické stránky.

7 HTTP servery

Kromě klasického zástupce HTTP serverů jako je Nginx se dnes začínají používat zabudované servery v programovacích jazycích nebo servery na nich postavené.

Například u JavaScriptu se využívá Node.js v případě generování šablon aplikace pomocí SSR. u PHP je ukázkovým příkladem ReactPHP.

7.1 Nginx

Nginx je jedním z nejznámějších HTTP serverů. Byl vyvinut Igorem Sysoevem a první verze vyšla v roce 2004.

Výhodou Nginxu je rychlost, nízké požadavky na serverové prostředky, jednoduchá a podrobná konfigurace a flexibilita. Kromě HTTP serveru pro webové aplikace se Nginx používá i jako reverzní proxy server. Je dostupný ve verzi zdarma a komerční licenci. Komerční licence je vývojově napřed a obsahuje funkcionalitu navíc (viz [w-35]).

7.2 Node.js

Node.js v sobě má zabudovaný HTTP server. Tento server je velmi jednoduché zprovoznit. Stačí vytvořit libovolně nazvaný.js soubor do kterého se vloží inicializace http serveru (viz [w-36]).

```
const http = require("http");
const requestListener = (req, res) => {
    res.writeHead(200);
    res.end("Ahoj!");
};
const server = http.createServer(requestListener);
server.listen(8080);
```

Obrázek 19: Ukázka inicializace Node.js HTTP serveru

Nad Node.js http serverem jsou pak postavené frameworky jako je express.js. Tento framework umožňuje jednodušeji vytvářet routy, které poslouchají na všechny typy requestů jako je post, put a get.

```
const express = require("express");
const app = express();
const port = 3000;

app.get("/", (req, res) => res.send("Ahoj!"));
app.listen(port, () => {
    console.log("Server poslouchá na http://localhost:" + port);
});
```

Obrázek 20: Ukázka inicializace Express.js serveru

7.3 PHP

PHP má stejně jako Node.js zabudovaný HTTP server. Tento server je opět možné spustit jedním příkazem.

Rozdíl oproti Node.js Http serveru je, že se konfigurace provádí přímo v CLI a konfiguraci PHP je možné upravit přes .ini soubor. Uvnitř a uvnitř index.php souboru už může být jakýkoliv kód. Vývojáři sem zpravidla dávají nastavení rout, které na základě dotazu vrací patřičný obsah.

Níže je uvedena ukázka spuštění PHP web serveru pomocí systémové konzole. Nejdříve je potřeba CLI přesunout do složky se souborem index.php a následně dalším příkazem spustit server (viz [w-3]).

```
cd ~/public_html
php -S localhost:8000
```

Obrázek 21: Ukázka spuštění PHP web serveru

7.4 IIS Express

IIS (Internet Information Services) Express je rozšíření do Visual Studia vyvíjené Microsoftem. Toto rozšíření vzniklo, protože vývojáři potřebovali „zkombinovat“ benefity z ASP.NET vývojového serveru a IIS Web Serveru (viz [w-68]).

Toto rozšíření je součástí instalace Visual Studia a umožňuje vývojáři spustit webový projekt napsaný např. v ASP.NET pomocí jednoho kliknutí. V základním nastavení běží web na localhostu na portu 8080. Server se dá ale nakonfigurovat a lze do něj nainstalovat např. i rozšíření pro PHP (viz [w-67], [w-66]).

7.5 Ostatní

Pro výše zmíněný Nginx může být vhodnou alternativou Apache nebi IIS.

Zajímavou alternativou pro PHP HTTP server je ReactPHP. ReactPHP je knihovna pro událostmi řízené programování. Konfigurace vypadá podobně jako u PHP HTTP serveru. Rozdíl je v tom, že ReactPHP je asynchronní, a tedy během zpracovávání jednoho dotazu neblokuje další příchozí dotazy. Celkově je tato knihovna složená z několika komponent. Tyto komponenty se dělí na jádro, síťové, protokolové a utility komponenty.

Níže je uvedena oficiální ukázka inicializace ReactPHP serveru (viz [w-37]).

```php
<?php

    $loop = React\EventLoop\Factory::create();
    $server = new React\Http\Server(
        function (Psr\Http\Message\ServerRequestInterface $request) {
            return new React\Http\Response(
                200,
                array('Content-Type' => 'text/plain'),
                "Ahoj!\n"
            );
        }
    );
    $socket = new React\Socket\Server(8080, $loop);
    $server->listen($socket);

    echo "Server běží na http://127.0.0.1:8080\n";
    $loop->run();
```

Obrázek 22: Ukázka inicializace ReactPHP serveru

8 Databázové systémy

Ve světě databází se v poslední době začínají kromě klasických lokálně instalovatelných SQL a NoSQL databází vyskytovat serverless databáze. Tento typ databází vývojářům umožňuje rychle vytvořit novou databázi a šetří práci s jejich správou a škálováním.

8.1 MySQL

MySQL je tradičním zástupcem relačních databází. Vytvořena byla Davidem Axmarkem a Michaelem Wideniusem v roce 1994. Aktuálně tuto databázi vlastní a vyvíjí Oracle.

Ryan Stephens, Ron Plew a Arie Jones [op-4] o MySQL napsali *„MySQL je relační databázový systém, který byl navržen s ohledem na rychlost, všestrannost a provozní spolehlivost. Řada jeho schopností vznikla jako přímý důsledek automaticky potvrzovaných operací vkládání, aktualizace a mazání. Díky tomu již nejsou potřebné segmenty pro rollback ani ostatní dolaďovací mechanismy přítomné v jiných implementacích jazyka SQL. Ve výbavě najdete nejrůznější bezpečnostní nástroje včetně možnosti přiřazovat oprávnění a práva.“*

8.2 FaunaDB

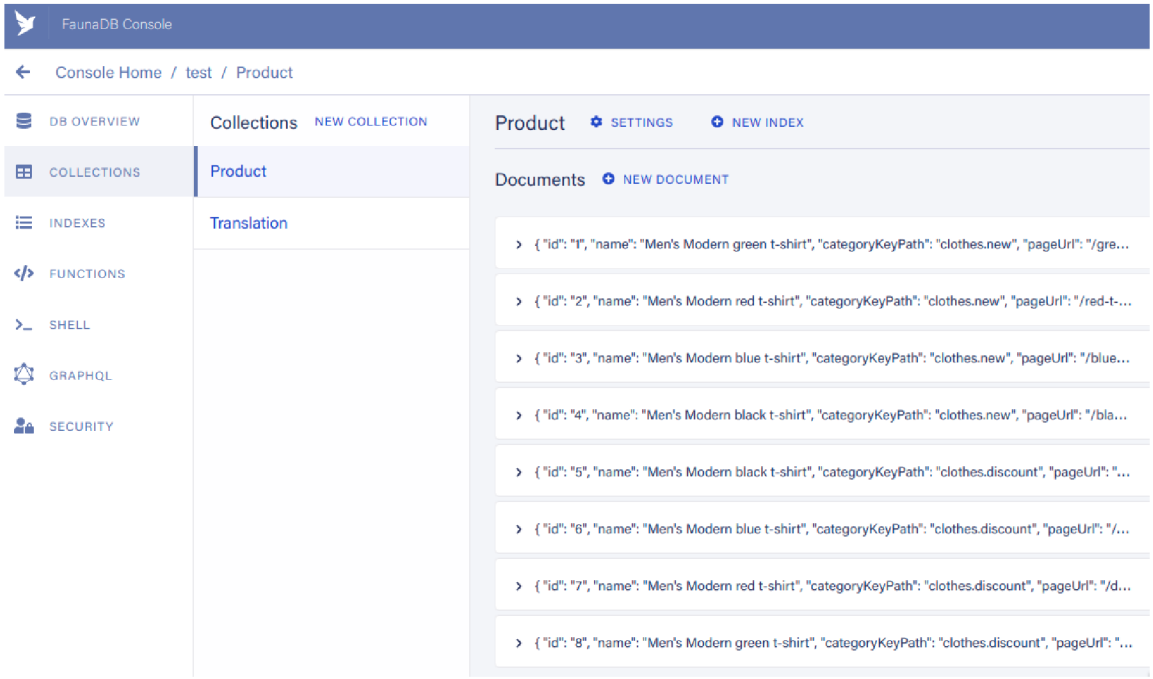
FaunaDB je moderní NoSQL, dokumentově orientovaná, konzistentní, serverless databáze založená Evanem Weaverem a Mattem Freelsem (viz [w-38]).

Oproti například výše zmíněnému Postgresu tato databáze automaticky řeší škálování napříč datacentry po celém světě, aby se snížila doba odezvy. Tuto databázi také není možné stáhnout a nainstalovat na vlastní server. Je potřeba se zaregistrovat a následně s ní pracovat na dálku.

Ke komunikaci s touto databází se využívá GraphQL API (viz graphql.org). Namísto tabulek tato databáze využívá kolekce dokumentů. Schémata kolekcí se vytváří naimportováním GraphQL schémat nebo pomocí dotazů přes API (viz [w-39]).

Pro nastavení rolí se v této databázi využívají bezpečnostní tokeny, které je potřeba vygenerovat a vložit dovnitř dotazu na API. Momentálně je možné vygenerovat Server a Admin token. Tyto tokeny se liší v oprávněních pro čtení, zápis a přístup k datům.

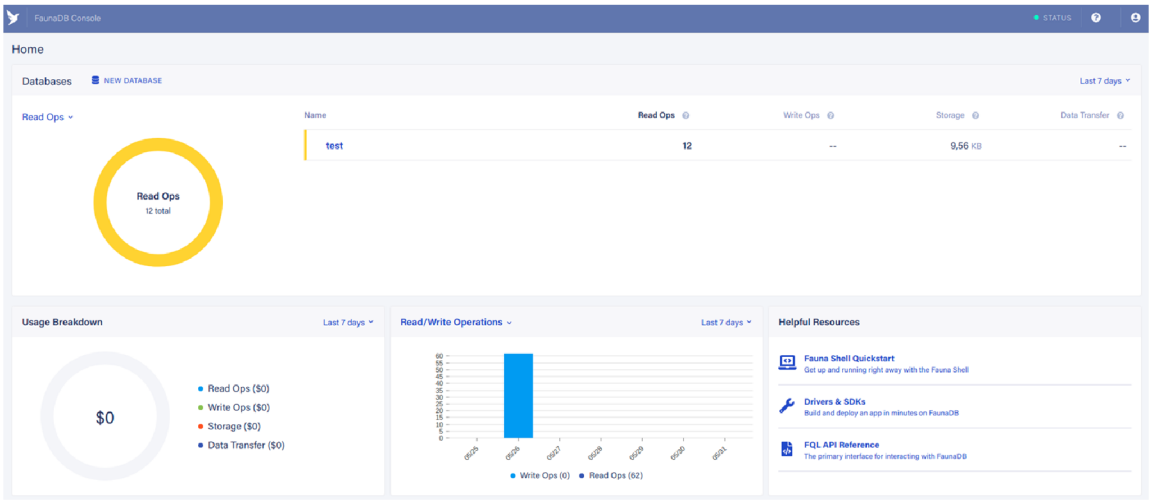
Vedle GraphQL FaunaDB poskytuje také FQL pro složitější dotazy, funkce a pokročilé transakce. FaunaDB má i připravené ovladače (SDK) pro FQL v jazycích například C#, Go, Pythonu a JavaScriptu. Díky tomu je možné psát FQL dotazy v jazyce, ve kterém je aplikace vytvářena (podobně jako v Entity Frameworku [viz docs.microsoft.com/cs-cz/ef] a LINQ [viz docs.microsoft.com/cs-cz/dotnet/csharp/linq]). Tyto ovladače exportují FQL funkce, které vypadají jako funkce dostupné ve zvoleném používaném programovacím jazyce. Tyto funkce lze libovolně skládat dohromady a vytvářet tak dotazy do FaunaDB.



Obrázek 23: Ukázka rozhraní stránky s kolekcemi ve FaunaDB administraci

FaunaDB oproti jiným databázím na hlavní stránce zobrazuje statistiky, které ukazují počet zapisovacích i čtecích requestů, velikost uložených data objem přenesených dat.

Cena za používání FaunaDB je plovoucí a záleží na počtu právě již zmíněných requestů do databáze, velikosti a objemu přenesených dat.



Obrázek 24: Ukázka rozhraní hlavní stránky FaunaDB administrace

8.3 Ostatní

Alternativou NoSQL databáze FaunaDB může být například MongoDB (viz mongodb.com/2). Tuto databázi je možné jednak nainstalovat a také využívat skrze MongoDB cloud. V případě cloudu odpadá starost o servery. Zároveň je možné začít na programu Free, který poskytuje až 5 GB uložiště zdarma. u placeného programu se platí na základě přenesených dat, velikosti RAM a velikosti uložiště.

Vhodná alternativa pro MySQL je PostgreSQL (viz postgresql.org), který může být stejně jako MySQL nainstalován lokálně. Další možnou alternativou je také SQL Server Express od Microsoftu. Pro studenty může být vhodný i přímo na Microsoft Azure. Zde mají pro studenty spoustu možností na pokusy zdarma (viz azure.microsoft.com/free/students).

9 GIT

Git (viz git-scm.com) je verzovací systém vytvořený Linusem Torvaldsem. Původně to byl nástroj určený pro vývoj jádra Linuxu.

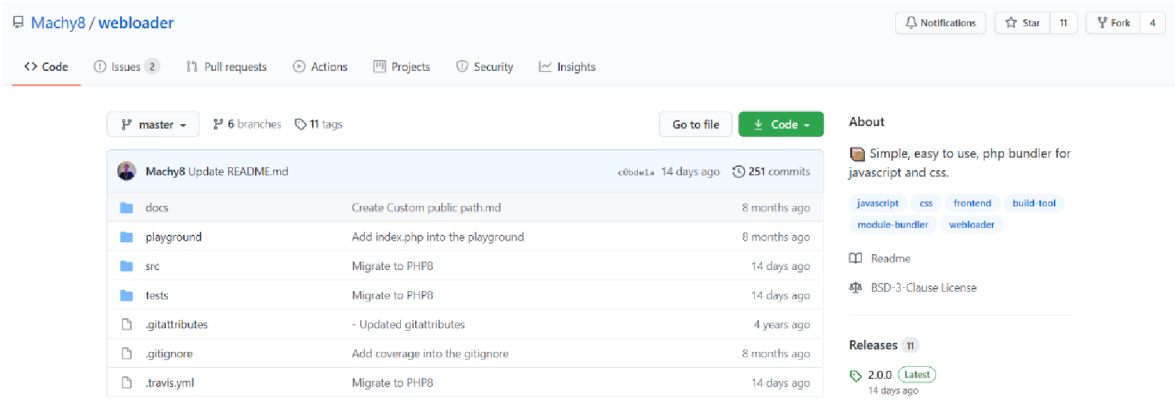
V současnosti je Git používán na různorodých projektech. Tento nástroj umožňuje projekt rozdělovat (větvit) podle vyvíjených částí, verzovat, přeskládávat kód z vedlejších větví na hlavní větev (master) a navěšovat posluchače na události jako je commit, rebase apod. (viz [w-40]).

Git je využíván i firmou Microsoft při vývoji operačního systému Windows. Repozitář Windowsu má ale velikost 270 GB a počet souborů přesahuje 3,5 milionu. Git nebyl na takto veliké repozitáře stavěný. Například příkaz „git status“ běží po dobu 10 minut. Microsoft proto vyvinul GVFS, který sice soubory vypíše ale stáhne je až v momentě, kdy jsou otevřené nebo se s nimi pracuje. Díky tomu se pracuje jen s využívanými soubory a trvání příkazu „git checkout“ se tak zkrátili ze 3 hodin na 30 vteřin (viz [w-41]).

10 Prostory pro uložení projektu

10.1 GitHub

GitHub je často používaná služba pro správu Git repozitářů. Autorem je Chris Wanstrath a služba byla spuštěna v roce 2008. V roce 2018 jej koupil a nadále provozuje Microsoft, a proto distribuuje přes něj zdrojové kódy některých svých projektů a technologií (např. .NET Core). Veřejné repositáře a projekty je zde možné provozovat zdarma. Pokud chce vývojář použít GitHub pro soukromé účely, musí se řídit oficiálním ceníkem (viz github.com/pricing).



Obrázek 25: Ukázka stránky s repozitářem ve službě GitHub

V GitHubu je možné nalézt správu pull requestů a projektů. Dále zde je možnost vytvářet issue[17], stránky s dokumentací a nedávno byla přidána i možnost akcí[18] (alternativa pro například službu Travis CI).

V repozitáři má programátor možnost vidět počet commitů, větví, vydání, přispěvatelů (tzv. „kontributorů“), prostředí a v neposlední řadě licenci. Všechny tyto věci je možné rozkliknout a podívat se na detailní informace (viz [w-42]).

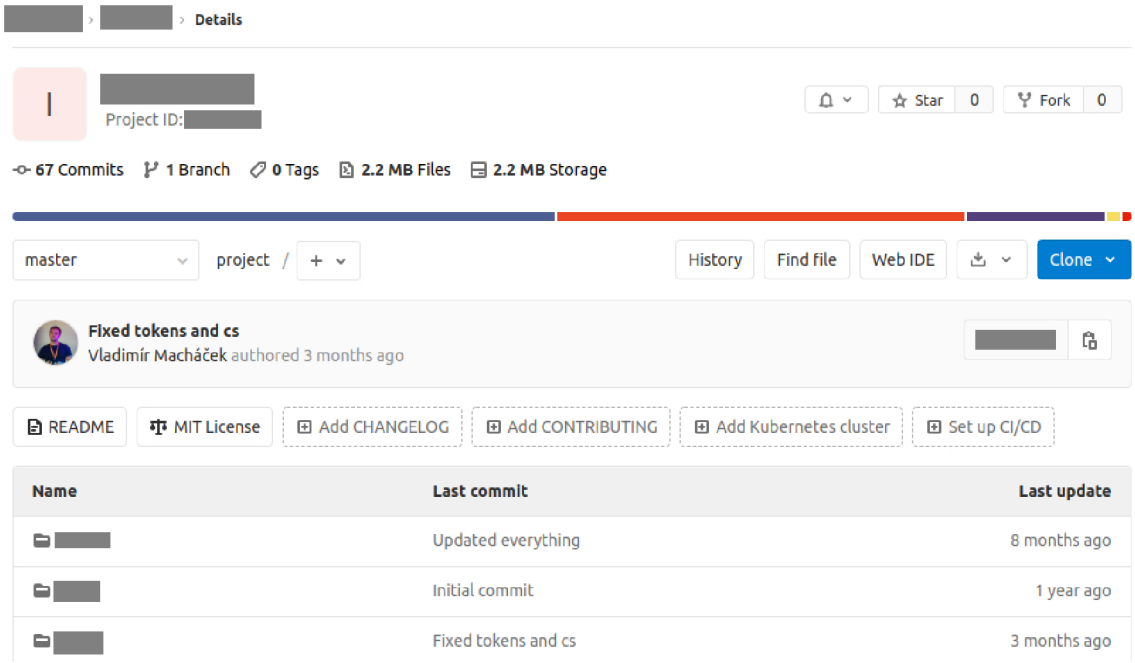
I když je GitHub primárně pro správu Git repozitáře, tak umožňuje i spouštět statické stránky. Tato služba se jmenuje GitHub Pages. Stránky jsou načítány ze zvoleného branche (větve kódu) a spuštěny na doméně priklad.github.io. Pro vytvořenou stránku je možné nastavit i vlastní doménu (viz [w-43]).

[17] Issues slouží pro přidávání poznámek, úkolů k danému repozitáři nebo nahlášení chyb ve funkčnosti kódu (viz [w-57]).
[18] GitHub akce umožňují např. spouštět testy a události, které se mají provést při každé změně poslané do repozitáře (viz [w-58]).

GitHub oproti níže zmíněnému GitLabu nabízí i desktopovou aplikaci, díky které odpadá nutnost spravovat repozitář ve webovém rozhraní. Zároveň je možné se v některých editorech na GitHub napojit a změny provádět pomocí kliknutí uvnitř editoru (viz [w-59]).

10.2 GitLab

GitLab je stejně jako GitHub služba pro správu Git repozitářů. Spuštěn byl v roce 2011 a vytvořil jej Dmitriy Zaporozhets.



Obrázek 26: Ukázka stránky repozitáře ve službě GitLab

Od GitHubu se liší hlavně uživatelským rozhraním, propracovanějším CI/CD, jinými možnostmi v nastavení práv uživatelů v týmu a lepším importem projektů z jiných služeb a repozitářů.

GitLab je na rozdíl od GitHubu možné nainstalovat na vlastní server a následně nakonfigurovat dle vlastních potřeb. Vývojáři tento přístup využívají z důvodu uložení kódu svých aplikací na vlastních zařízeních (viz [w-44]).

10.3 Ostatní

Obdobné služby jako jsou výše zmíněných jsou Bitbucket (viz bitbucket.org) a Azure DevOps (viz azure.microsoft.com/cs-cz).

11 Prostory pro provoz a spuštění webové aplikace

Webové aplikace lze dnes spustit různými způsoby a od klasických web hostingových řešení se ušla velká cesta.

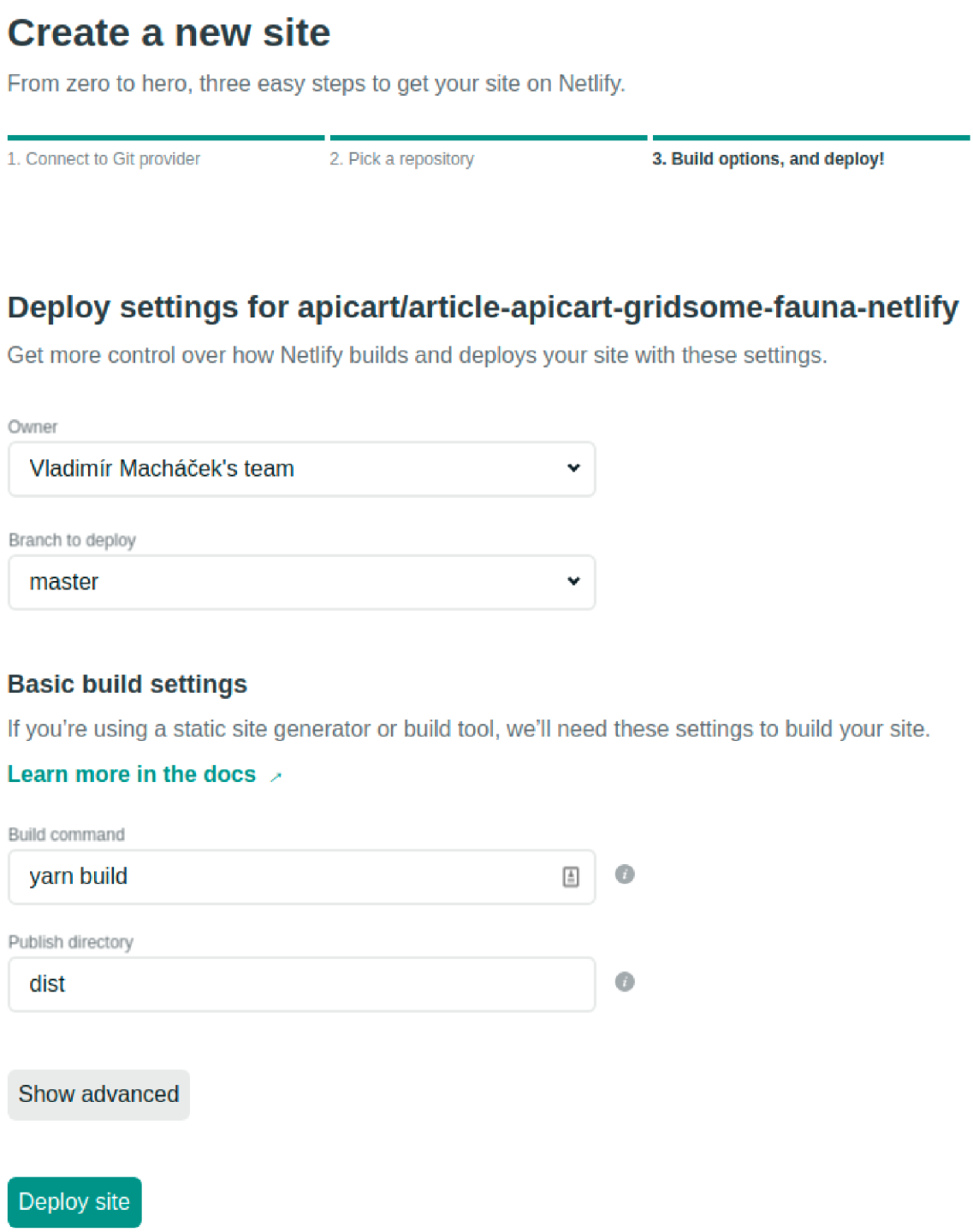
Je možné spustit aplikace v cloudu pomocí přístupu serverless, kdy rozsah využitého hardware pro danou aplikaci si poskytovatel hostingu řeší sám a vývojář pouze platí za využitý výkon a přenesená data.

Také se dnes dají využívat hostingy pouze pro statické stránky s možností využití serverless funkcí. Tedy lze mít stránku celou statickou a pouze pro určité části využívat dynamické funkce.

11.1 Netlify

Netlify je platforma sloužící ke statických a lehce dynamických stránek založená roku 2014.

Spuštění stránky je jednoduché a je složeno ze 2 kroků. Nahrání projektu nebo nastavení adresy ke Git repozitáři a nastavení procesu vygenerování a deploye aplikace, který lze ve své podstatě úplně přeskočit, pokud potřebujete web pouze spustit. Stránku je tak možné spustit během 5 minut a do určitého vytížení zdarma. V základním nastavení aplikace běží na doméně priklad.netlify.app. Tuto doménu je možné změnit v nastavení (viz [w-45]).

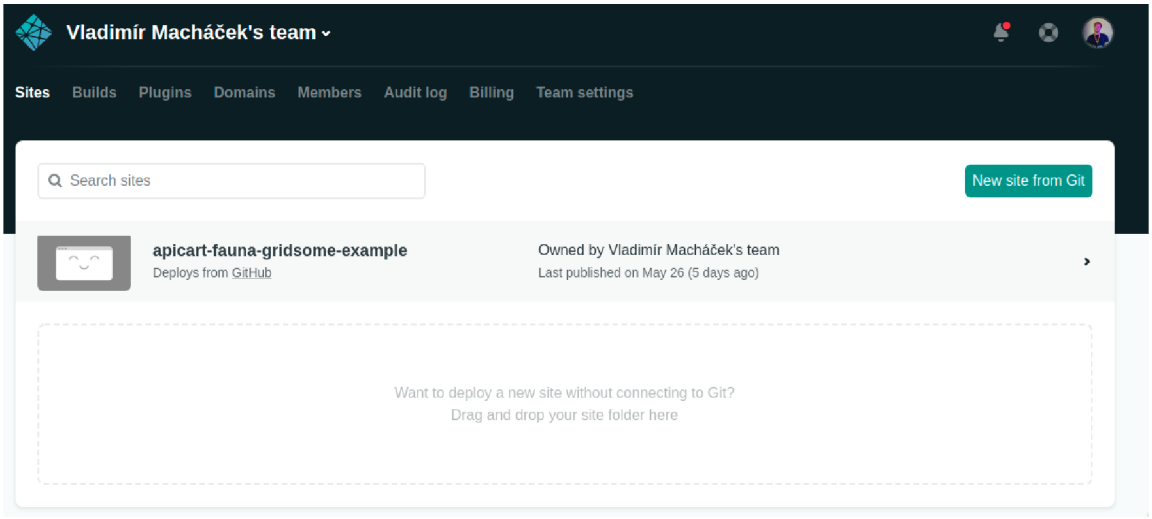


Obrázek 27: Ukázka spuštění webové aplikace na Netlify

Build aplikace je možné ovlivňovat environmentálními proměnnými a je možné instalovat JavaScriptové balíčky, spouštět příkazy pro například vygenerování statických stránek apod. Výsledná stránka také automaticky dostane SSL certifikát, který je automaticky obnovován.

Po nahrání aplikace je v administraci možné prohlížet log z buildu a spouštění aplikace, nastavit doménu a certifikát nebo přidávat a odebírat správce projektu.

U Netlify vývojář platí za velikost přenesených dat za měsíc, počet členů v týmu a počet paralelně běžících buildů aplikace. Pokud se vývojář vejde se svými nároky do free programu, tak neplatí nic (viz [w-46]).



Obrázek 28: Ukázka hlavní stránky Netlify administrace

11.2 Vercel

Vercel, původně Zeit je platforma pro deploy bez stavových aplikací používající přístup Serverless a stejně jako u Netlify se výsledná cena za poskytnuté služby odvíjí od vytížení aplikace (viz [w-47]).

Rozdíl mezi Vercelem a Netlify je ten, že na Vercelu je možné spouštět i dynamické aplikace běžící například na PHP nebo Node.js.

Aplikace jsou zde bez stavové, a tedy s dalším nahráním aplikace (deployem) jsou data uložená na aktuálně běžící instanci aplikace smazána.

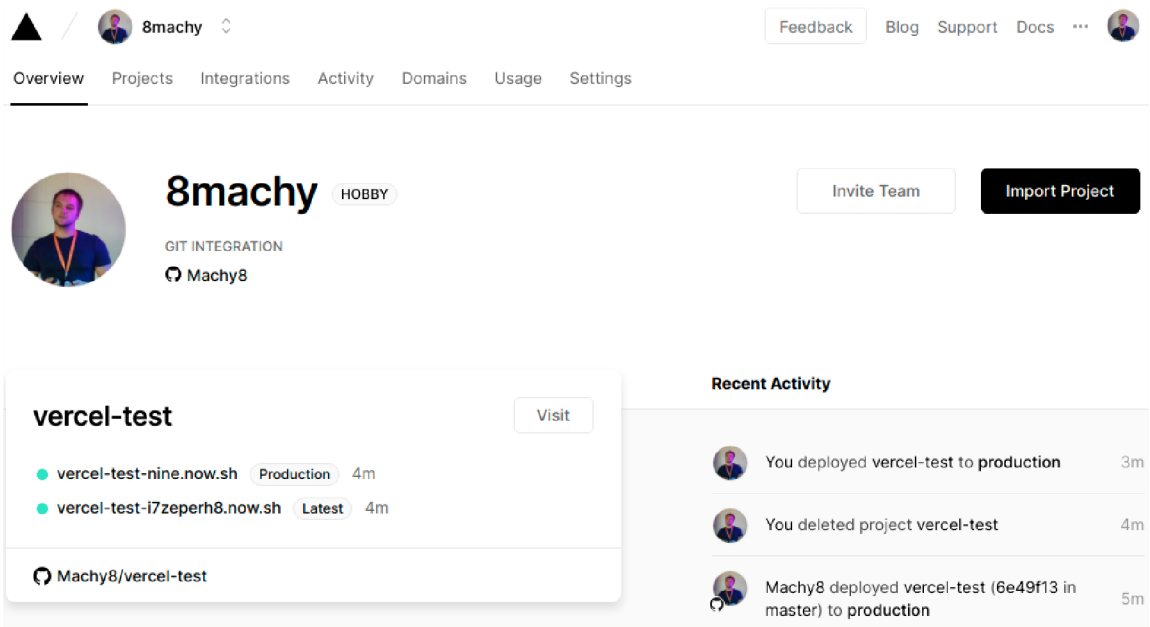
Spuštění aplikace vypadá obdobně jako na Netlify. V základu Vercel spustí stránku jako statickou. Pro spuštění stránky v nějakém prostředí je potřeba přidat vercel.json. V tomto konfiguračním souboru je možné nastavit prostředí a jeho parametry.

Na výběr je z několika prostředí. Jsou zde dostupná oficiální prostředí pro Node.js, Go, Python a Ruby a pak komunitní prostředí například pro PHP a Rust (viz [w-48]).

```
{
    "functions": {
        "api/*.php": {
            "runtime": "now-php@0.0.10"
        }
    }
}
```

Obrázek 29: Ukázka konfiguračního souboru pro službu Vercel

Následně je možné naimportovat repozitář z GitHubu, GitLabu, anebo Bitbucketu. Během konfigurace spuštění aplikace může vývojář stejně jako u Netlify nastavit parametry pro prostředí a příkazy, které se mají spustit při vytváření aplikace.



Obrázek 30: Ukázka administrace služby Vercel

11.3 Webhostingy, VPS a dedikované servery

Webhostingy jsou jednou z nejstarších možností, kde je možné provozovat webové stránky. Pro spuštění webové stránky na webhostingu většinou nemusí vývojář nic konfigurovat a stačí pouze nakopírovat obsah na poskytnuté uložiště. Nahrání souborů je většinou možné pomoci FTP, anebo webového rozhraní poskytnutého daným webhostingem. Webhostingy jsou vhodné pro menší projekty. Jsou jednoduché na používání. Většinou mají nízké parametry, co se týče výkonu, jako frekvence procesoru a velikost RAM. U webhostingů je také většinou přesně daný software a není možné si

upravovat nastavení prostředí. Výhodou webhostingů je i cena, která je mnohdy nízká a může se pohybovat kolem pouhých 500 korun českých za rok (viz [w-49]).

Pokud programátor potřebuje větší flexibilitu a výkon, tak může zvolit VPS[19] nebo dedikovaný server. Tyto dvě možnosti jsou náročnější na údržbu, jsou dražší a pro správu je již potřeba určitých znalostí týkajících se správy serveru, což u výše zmíněných služeb není podmínkou. Na druhou stranu u VPS a dedikovaného serveru má programátor možnost přesně nainstalovat co potřebuje. VPS je vhodné v momentě kdy nedostačuje webhosting a je potřeba větší flexibilita. VPS může běžet na jednom serveru s více instancemi a pokud si vývojář nepřiplatí za vyhrazený výkon, tak pak může být výkon zakoupeného VPS ovlivněn dalšími instancemi. Pokud je zvolen dedikovaný server tak vývojář dostane celý server pro sebe a výkon mu není odebírán dalšími instancemi. Dedikované servery jsou ale dražší a složitější na údržbu než VPS (viz [w-50]).

11.4 Ostatní

Dalšími platformami, kde je možné web spustit jsou Amazon Web Services, Google Cloud, Microsoft Azure a Digital Ocean. Tyto platformy vyžadují ke své správě již patřičné znalosti a nejsou přímo vhodné pro začátečníky

Všechny čtyři cloudové služby zároveň nabízí i slevy a speciální programy pro studenty. Amazon Web Services nabízí Aws Educate program, který poskytuje vzdělávací materiály, free program na Amazon nástroje a seznam pracovních pozic (viz aws.amazon.com/education/awseducate/students). Pokud je škola zapojena do vzdělávacího systému od Google (viz edu.google.com/programs/faqs/teaching-and-learning-credits), tak může student zažádat o kredity na provoz služeb (viz cloud.google.com/billing/docs/how-to/edu-grants). Microsoft Azure aktuálně v době psaní této práce (duben 2021) nabízí kredity v hodnotě $100 a vývojářské nástroje zdarma s možností registrace bez platební karty (viz azure.microsoft.com/cs-cz/free/students). Podobnou nabídku má i Digital Ocean pro studenty, kteří se

[19] Virtuální privátní server je flexibilnější varianta hostování webu. Jedná se o virtualizovaný typ serveru, který má přiřazený zvolený výkon a většinou běží paralelně s více virtuálními servery na jednom fyzickém serveru (viz [w-50]).

zaregistrují na GitHubu do vzdělávacího programu (viz digitalocean.com/github-students).

Výhodou je škálovatelnost a automatická správa. Vývojář se nemusí starat o hardware a software, ani přenášení výkonu mezi servery či zálohování dat. Nevýhodou je mnohdy složitost a na správu aplikací využívající cloudová řešení je zapotřebí vyhrazený vývojář.

U těchto služeb se povětšinou platí například za velikost přenesených a uložených dat a za dobu využívání a vytížení CPU či RAM. Cena je díky tomu flexibilní a záleží na vytížení webové aplikace. Na druhou stranu to může způsobit také problémy v případě, kdy například špatně naprogramovaná aplikace posílá velké množství dat nebo vytěžuje CPU či RAM. V takovém případě může být výsledná částka za používání těchto služeb poměrně vysoká (viz [w-51]).

Alternativy, které jsou dostupné zdarma aktuálně k roku 2021 mohou být například hosting na webzdarma.cz (viz webzdarma.cz/cenik), endora.cz (viz endora.cz/cenik) nebo infinityfree.net (viz infinityfree.net).

12 Požadované technologie na trhu práce v ČR

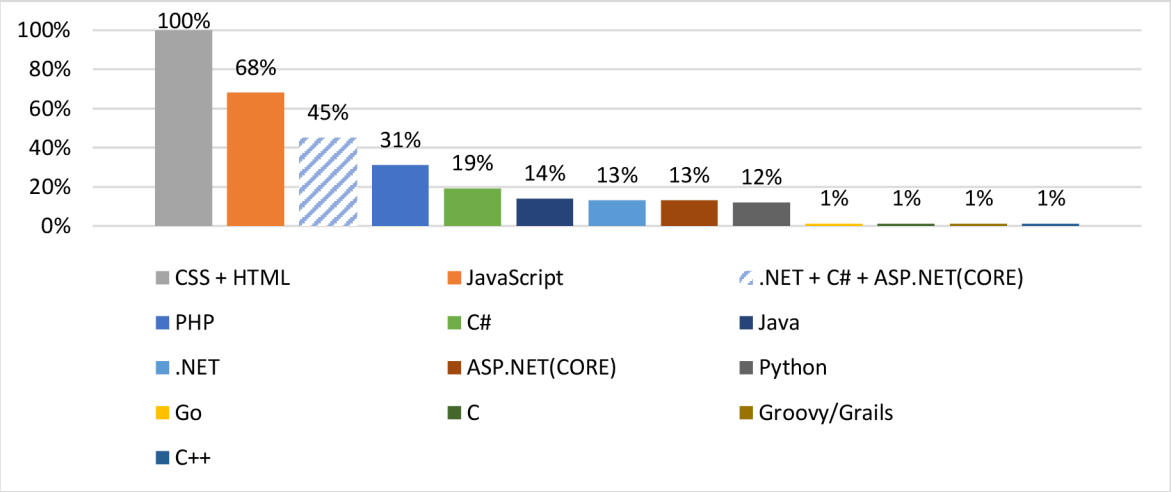
U výzkumu nebyl brán zřetel na odvětví ani velikost firmy. Zároveň také nebylo rozlišováno, jaká úroveň uchazeče je požadována. Informace byly sbírány z běžně dostupných pracovních portálů, ale také přímo ze stránek firem. Hodnoty v grafech jsou uvedeny v procentech a jsou vypočítány z celkového počtu projitých inzerátů (viz níže).

- **Počet prozkoumaných pracovních inzerátů:** 100.
- **Použitá klíčová slova pro vyhledávání pracovních inzerátů:**
    - webový vývojář, vývojář webových stránek, programátor webových stránek, frontend vývojář, full stack vývojář, UX designer, web developer.
- **Počet použitých pracovních portálů a stránek s inzeráty:** 12.
- **Použité pracovní portály:**
    - jobs.cz, cz.indeed.com, startupjobs.cz, jobs.tipsport.cz, alza.cz/kariera, czc.cz/volna-mista, spolecnost.vivantis.cz/kariera, pearshealthcyber.cz, kariera.drmax.cz/volne-pozice, slevomat.cz/prace, zivotvrohliku.cz/volne-pozice, airbank.cz/pracujte-u-nas.

12.1 Inzerenti zahrnutých pracovních inzerátů

Inzerenti byly voleni namátkově z výše uvedených serverů s pracovními nabídkami. Odvětví zahrnutá v použitých inzerátech jsou například banky, online média, e-shopy s různými zaměřeními, poskytovatelé webhostingů, ale také firmy vytvářející weby na zakázku. Seznam inzerentů je abecedně seřazen (viz Příloha 1).

12.2 Jazyky pro vývoj webu

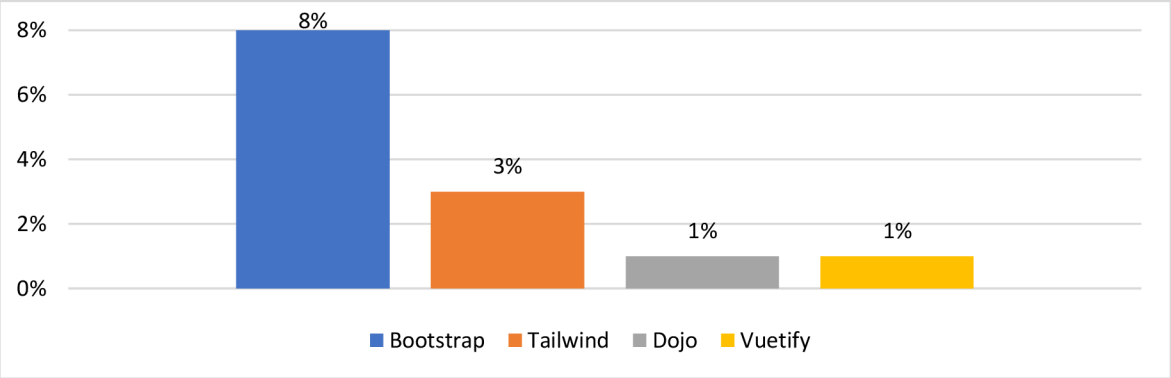


Obrázek 31: Přehled požadovaných jazyků pro vývoj webu v pracovních inzerátech

Z grafu výše jasně vyplývá, že každá firma v inzerátu požadovala alespoň minimální znalost značkovacích jazyků CSS+HTML. Nejpožadovanějším programovacím jazykem, který se vyskytl jako požadavek v 68 % inzerátů, je JavaScript. Nicméně v inzerátech nebylo uváděno, zdali JavaScript potřebují pouze pro frontend, nebo z důvodu používání např. Node.js. Pomocí JavaScriptu lze dnes vytvářet frontendová i backendová řešení, ale je také možné ho využít pro vytváření desktopových a mobilních aplikací. Díky tomu se stává čím dál více používaným a univerzálním jazykem.

Druhý nejčastější výskyt v inzerátech má jazyk PHP s 31 % a třetí C# s 19 %. Pokud bychom vzali v potaz to, že jazyky jako .NET, C#, ASP.NET a ASP.NET Core jsou podobné a jejich hlasy by se „sečetly“, tvořily by společně 45% podíl v požadovaných jazycích na trhu práce (viz fiktivní sloupec v grafu .NET, C#, ASP.NET (CORE)).

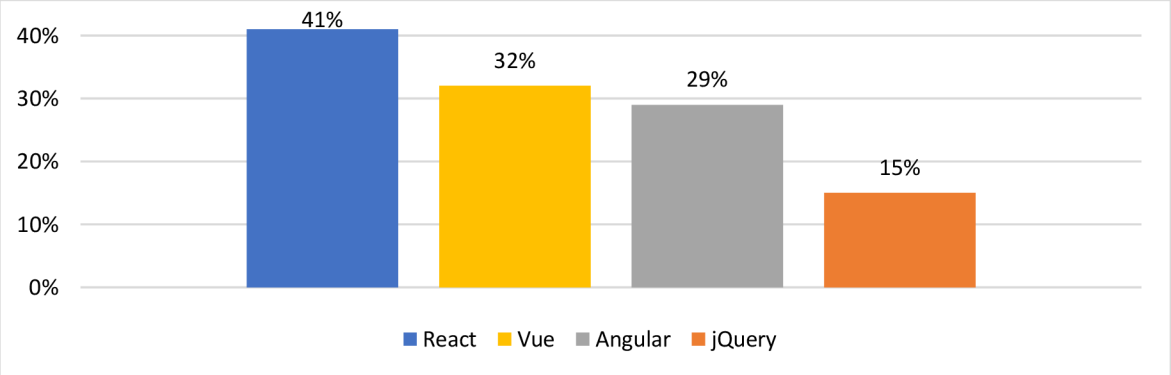
12.3 CSS Frameworky



Obrázek 32: Přehled výskytu CSS frameworků v pracovních inzerátech

Co se týče nejčastěji zmíněných CSS frameworků, tak zde byl na prvním místě Bootstrap. Byl zmíněn v 8 % inzerátů. Jako druhý byl Tailwind se 3 %. Důvod, proč Bootstrap převažuje je zřejmě kvůli rozsahu komunity, počtu napsaných návodů a nástrojích a šablon, který jsou na tomto frameworku postavené. Mezi další zmíněné frameworky pak patří Dojo a Vuetify, jež byly oba zmíněny v 1 % inzerátů. Je možné, že firmy tyto frameworky uvádí až při pracovním pohovoru nebo počítají s doškolením zaměstnanců po nástupu do zaměstnání.

12.4 JavaScript Frameworky



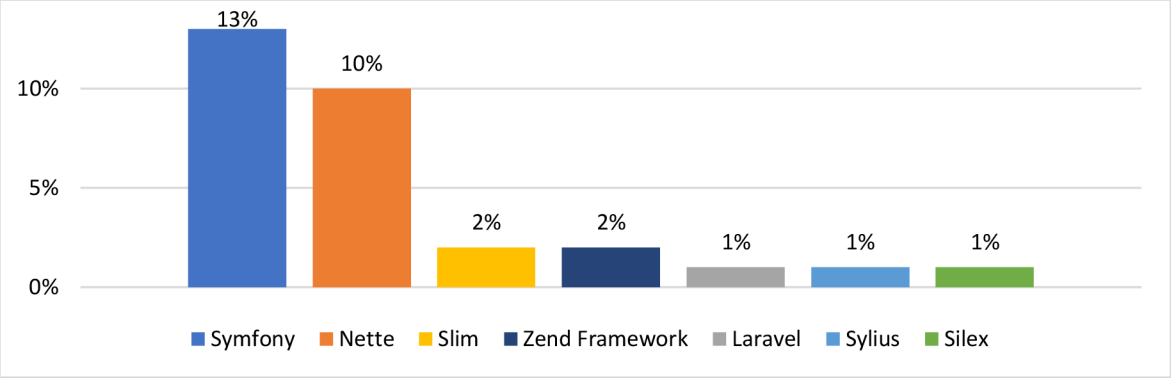
Obrázek 33: Přehled výskytu JavaScriptových frameworků v pracovních inzerátech

Mezi JavaScriptovými frameworky a knihovnami vede v počtu zmínění v inzerátech React s 41 % výskytu, druhým je Vue se 32 % a třetím Angular s 29 %. Posledním nejčastěji zmiňovanou knihovnou je jQuery s 15 %.

Důvodem, proč jQuery již není tolik vyžadovaná může být, že dnes již prohlížeče fungují víceméně stejně. Příkladem mohou být např. funkce pro získávání elementů na stránce

(viz caniuse.com/queryselector), manipulaci s třídami (viz caniuse.com/classlist) či iterace nad poli a prvky (viz caniuse.com/mdn-javascript_builtins_set_foreach) kvůli nimž se jQuery používala. Tyto funkce mají v době psaní této práce (duben 2021) všechny přes 95 % podporu napříč prohlížeči. Vývojáři jsou schopni si kompatibilitu napsaného skriptu ohlídat pomocí jiných nástrojů a kompilátorů, jako je např. Babel (viz kapitola 5.1) či rozšíření Typescript (viz kapitola 5.2) a z jQuery se tak na stránce stává pouze zbytečná závislost. Dalším důvodem je velmi pravděpodobně i to, že dnešní webové aplikace jsou složitější, hodně částí je dynamických a překreslovat velké stránky na základě změny dat pomocí jQuery je složité. Reactu, Vue i Angular jsou za tímto účelem přímo vyvinuty. Kód je přehlednější, a navíc pro ně existují i nástroje pro vykreslování šablon již na straně serveru. Firmy také mohou používat vlastní frameworky, které byť třeba některou z těchto technologií na pozadí používají, tak se k nim vývojář ani nedostane a používá syntaxi či průvodce firemního frameworku.

12.5 PHP Frameworky

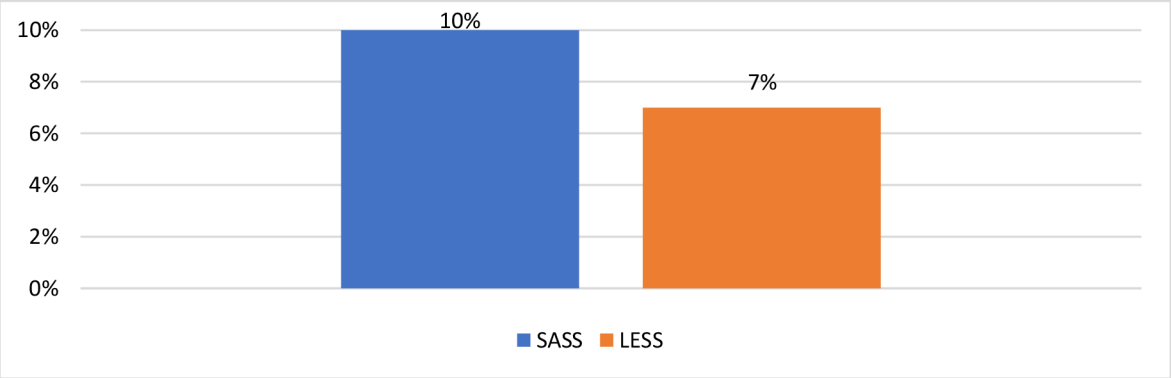


Obrázek 34: Přehled výskytu použitých PHP frameworků v pracovních inzerátech

V inzerátech byly nejčastěji zmiňovány frameworky Symfony s 13 % a Nette s 10 % výskytu. Ostatní zmiňované byly například Slim, Laravel, Silex, Sylius či Zend Framework. Nette v Česku vede bezpochyby díky české komunitě zahrnující fóra, chaty, diskuse a články. Symfony oproti Nette má výhodu v tom, že je celosvětově používanější. Mezi faktory, proč firmy spíše upřednostňují Symfony oproti Nette může být fakt, že seženou jak vývojáře z České republiky, tak ze zahraničí, a že oproti Nette má Symfony celkově větší komunitu, více vývojářů a více knihoven a nástrojů, které s touto knihovnou fungují nebo jsou na ni postavené. To, že je Symfony využíváno

celosvětově více vyplývá například ze statistik, které byly zveřejněny na webu Sitepoint.cz (viz sitepoint.com/best-php-framework-2015-sitepoint-survey-results).

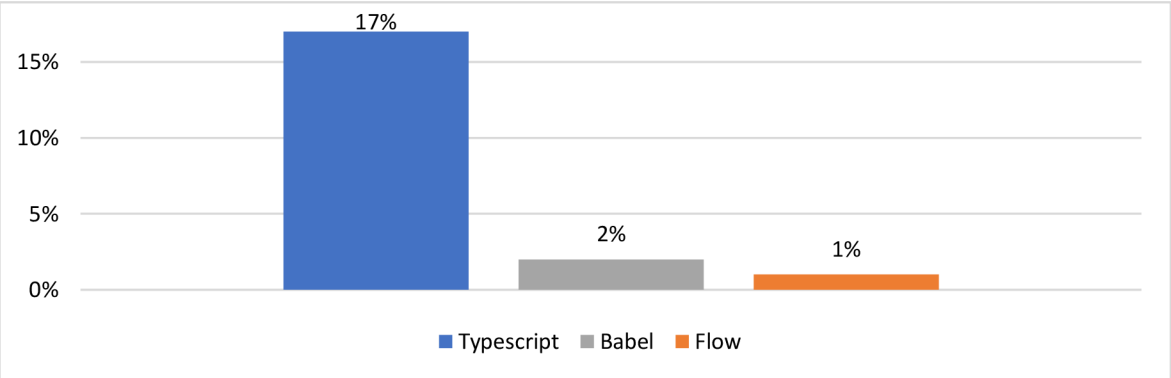
12.6 CSS preprocesory a postprocesory



Obrázek 35: Přehled výskytu CSS preprocesorů a postprocesorů v pracovních inzerátech

Vítězem mezi preprocesory a postprocesory v četnosti uvádění v inzerátech je SASS s 10 % výskytu. Druhým nejčastěji zmiňovaným preprocesorem je LESS se 7 % výskytu. Autoprefixer nebo PostCSS jako postprocesor nebyl uveden ani jednou. Důvodem může být, že oba preprocesory s tímto nástrojem již v základu spolupracují, a tedy se očekává, že vývojář s ním bude umět také pracovat (viz kapitola 4.1).

12.7 Rozšíření a kompilátory jazyka JavaScript



Obrázek 36: Přehled výskytu JavaScriptových kompilátorů a rozšíření v pracovních inzerátech

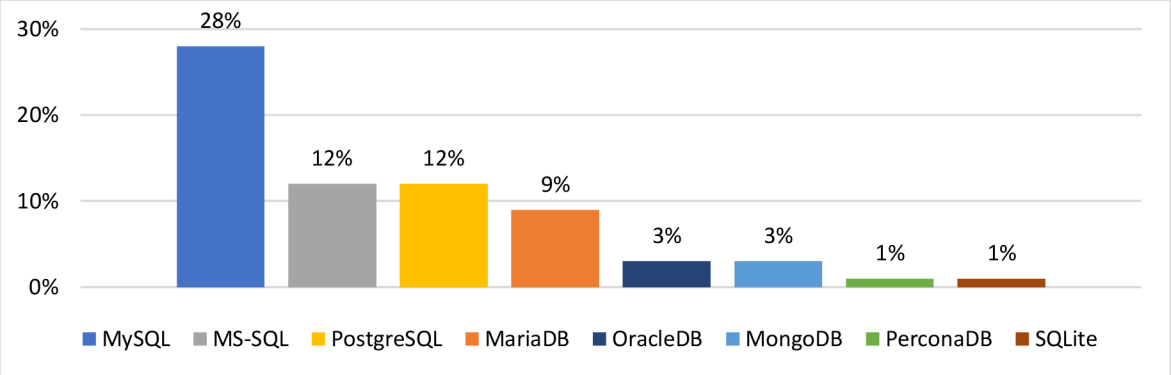
Nejvíce zmiňovaným rozšířením jazyka JavaScript byl Typescript se 17 % výskytu. Druhým byl Babel se 2 % a Flow s 1 %.

Typescript zde oproti alternativě Flow bezpochyby vede díky rozsáhlé komunitě a většímu počtu vývojářů. Určitě zde figuruje i fakt, že je vyvíjen vývojáři od Microsoftu.

Většina nejčastěji používaných editorů jako je Intellij Idea, Visual Studio či Visual Studio Code mají pluginy a rozšíření, který umí s Typescriptem pracovat.

I když má v grafu Babel pouze 2 %, tak pravděpodobně obdobně jako například PostCSS nebyl uváděn. Důvodem může být to, že s ním nově příchozí vývojáři vůbec nemusí přijít do styku. Napsaný kód se kompiluje na pozadí nebo až během deploye na základě konfigurace měnící se např. po několika měsících. Babelem je také možné zpracovat kód, který je napsaný v Typescriptu. Je tedy pravděpodobné, že je používán ve více než 2 % firem.

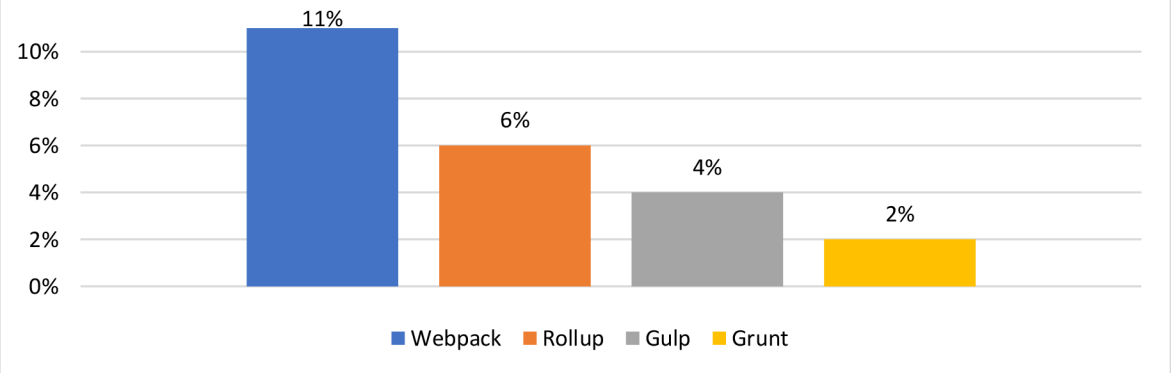
12.8 Databázové systémy



Obrázek 37: Přehled výskytu databázových systémů v pracovních inzerátech

Nejčastěji zmiňovanou databází v inzerátech byla MySQL s 28 %. Na druhém místě pak byly současně se stejným počtem zmiňovány PostgreSQL a MS-SQL (SQL Server) s 12 % výskytu. Ostatními zmíněnými databázemi byly MariaDB, MongoDB, OracleDB, PerconaDB a SQLite.

12.9 Balíčkovací nástroje

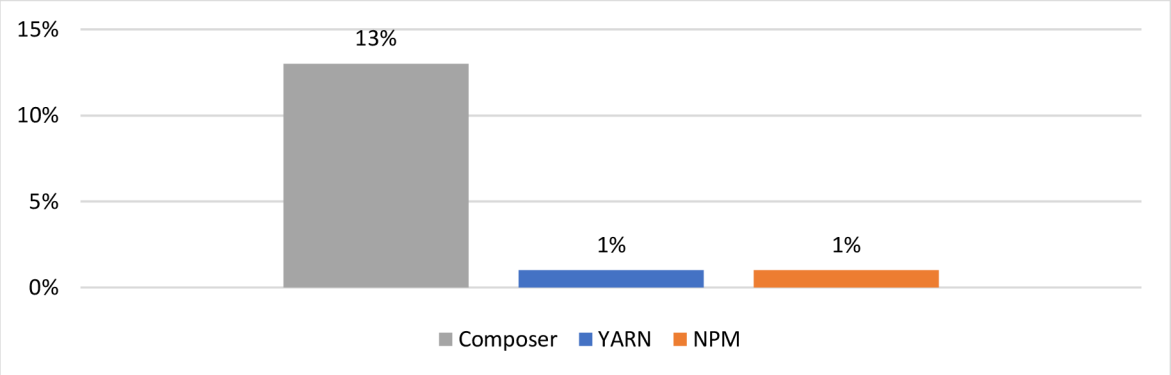


Obrázek 38: Přehled výskytu balíčkovacích nástrojů v pracovních inzerátech

Pro zpracování různých typů souborů jako je JavaScript, CSS, HTML, obrázky či fonty firmy nejčastěji požadovaly znalost Webpacku, který byl zmíněn v 11 % inzerátů. Druhým nejčastěji uváděným byl Rollup s 6 % výskytu. Další dva zmíněné byly Gulp (viz gulpjs.com) a Grunt (viz gruntjs.com). Důvodem, proč například nebyl uveden NuGet může být, že to firmy vzhledem k intuitivnosti nepovažovaly za nutné.

Rozdíl mezi Webpackem a Rollupem je primárně v konfiguraci a způsobu, jakým soubory zpracovávají. Webpack má také oproti Rollupu větší komunitu a rozsáhlejší nabídku pluginů. Také je na trhu déle a je více znám. Proto je pravděpodobně požadován častěji.

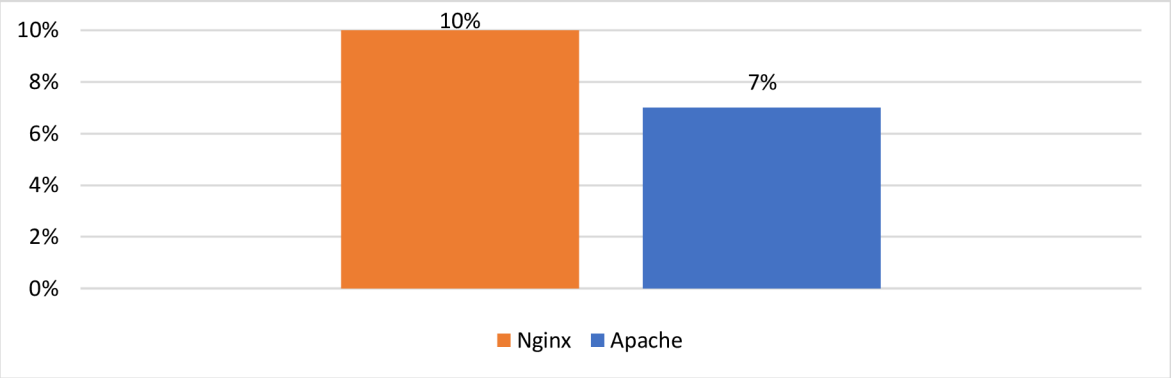
12.10 Nástroje pro správu a instalaci balíčků



Obrázek 39: Nástroje pro správu a instalaci balíčků v pracovních inzerátech

Nejuváděnějším nástrojem pro správu a instalaci balíčků a závislostí byl v inzerátech Composer a to v 13 % případů. Za ním byly uvedeny s 1 % Yarn a NPM. Důvodem, proč Composer má největší podíl může být, že nejvíce nalezených inzerátů požadovalo PHP a případně JavaScript. Chybějící NuGet pro .NET aplikace zde není pravděpodobně proto, že to firmy neuváděly, jelikož je součástí Visual Studia a nebylo jej tak zapotřebí uvádět zvlášť. Odpovídala by tomu i samotná statistika Composeru, Yarnu a NPM, protože tako mají dohromady malý počet v poměru na 100 inzerátů.

12.11 HTTP Servery

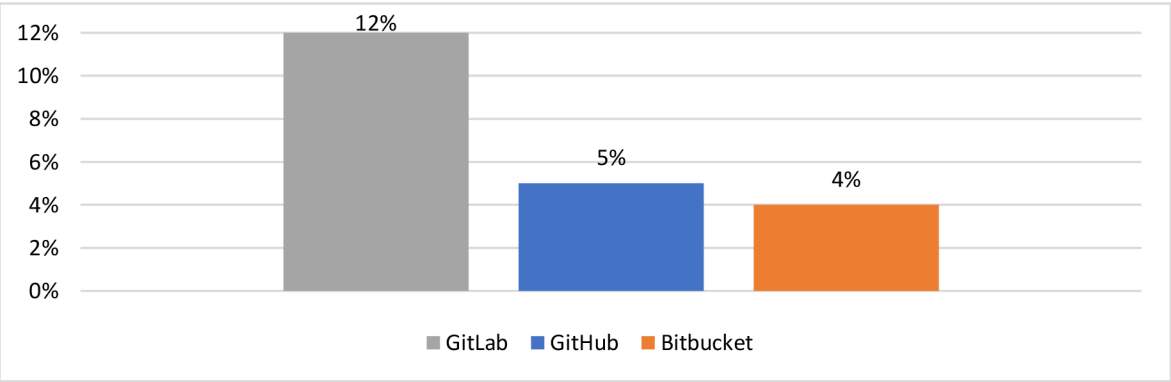


Obrázek 40: Přehled výskytu HTTP serverů v pracovních inzerátech

Během procházení pracovních inzerátů byly nalezeny pouze 2 HTTP servery. Jedním z nich je Nginx, který měl výskyt v 10 % a druhý Apache se 7 %. I když je Apache na trhu déle, Nginx získává na popularitě díky jednoduché konfiguraci a rychlosti, což je pravděpodobně důvod, proč je jeho znalost požadována častěji.

Protože se v inzerátech vyskytly pouze dva typy serverů, je možné, že je firmy neuvádí. Důvodem může být, že jeho správu zařizuje externí firma případně jiný zaměstnanec, nebo se k nim uchazeč vůbec ze začátku nedostane. Může to být důvod, proč se zde například vůbec nevyskytuje IIS server nebo Azure.

12.12 Prostory pro uložení projektu



Obrázek 41: Přehled výskytu prostorů pro uložení projektů v pracovních inzerátech

Mezi prostory pro uložení projektu byl nejčastěji zmiňován GitLab. Byl zmíněn ve 12 % inzerátů. Svou popularitu získává pravděpodobně tím, že je možné GitLab používat jak přímo hostovaný na GitLabu, tak u sebe na svých serverech. Díky tomu

mohou firmy mít svůj kód stále na svých uložištích[20] a spouštět GitLab například na výkonnějších[21] serverech (viz kapitola 10.2). Pokud firmy využijí GitLab na svých serverech, tak se také nemusí obávat nečekaného ukončení podpory nebo zvýšení cen provozu. Druhý nejčastěji zmiňovaný je GitHub, který byl zmíněn ve 5 % inzerátů se 4 % výskytu je poslední Bitbucket.

12.13 Další zmíněné nástroje

Další zmíněné technologie, které se v inzerátech vyskytovaly spíše unikátně, proto nebyly zařazeny do grafů jsou v Příloze 2. Technologie jsou seřazeny abecedně.

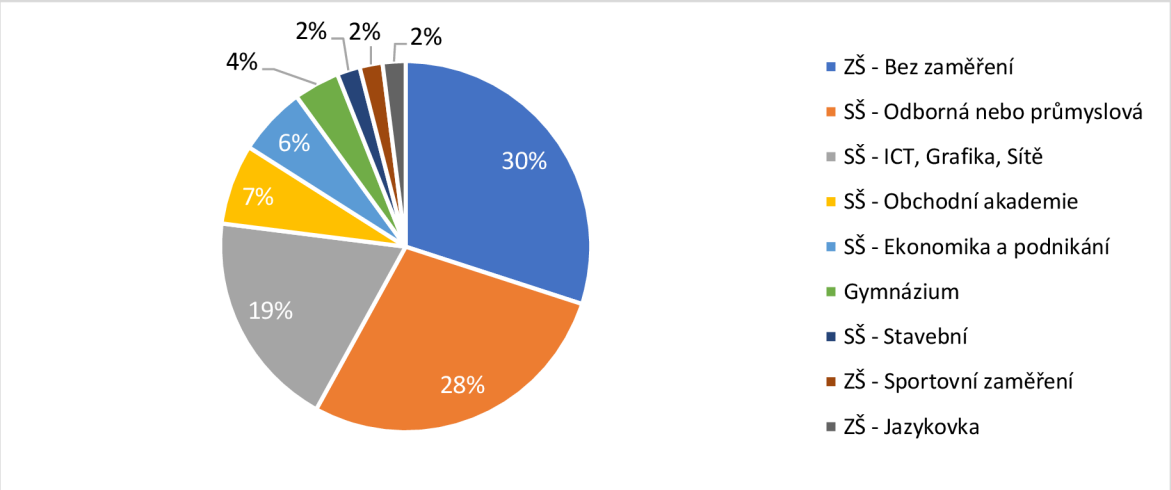
[20] Firmy se mohou obávat toho, že jim kód na vzdálených serverech může někdo ukrást nebo smazat. Proto pravděpodobně volí vlastní servery.

[21] I když služby GitLabu běží v Cloudu, tak některé operace mohou být pomalejší z důvodu zatížení serverů například útoky nebo operacemi pro ostatní klienty. Může to být další z důvodů, proč firmy spouští GitLab na svých serverech. Mají díky tomu výkon serverů pouze pro své potřeby a v některých případech to může být rychlejší.

13 Vyučované technologie na školách

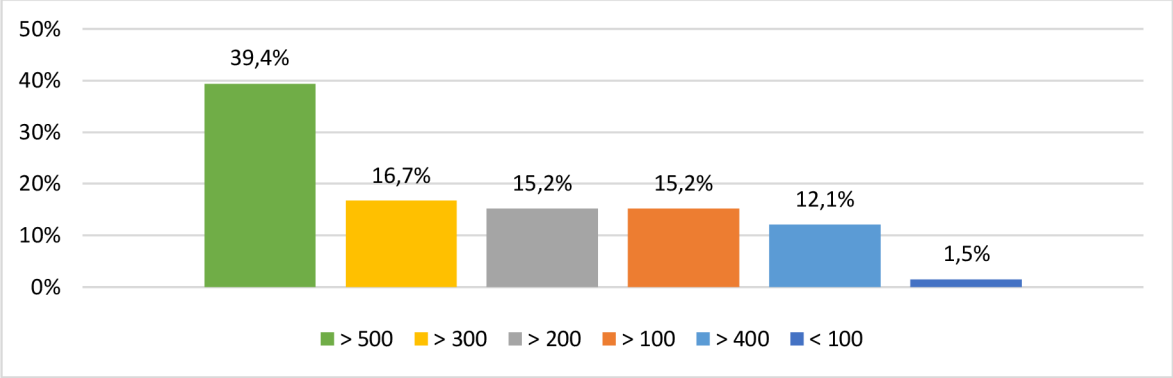
Informace o tom, co vyučují na základních, středních i vysokých školách byly získány pomocí rozeslaných dotazníků.

Celkem ze 113 rozeslaných dotazníků se jich vrátilo vyplněných 66 z 52 různých škol. Ze středních škol bylo získáno 46 odpovědí, ze základních škol 18 a z gymnázií 2. Z vysokých škol nebyl zodpovězen žádný dotazník. V grafech jsou mnohdy uvedena různá čísla počtu odpovědí. Odpovědi, pokud není uvedeno jinak, jsou brány z hlediska učitelů. Je to z toho důvodu, že na škole může být na jeden předmět více vyučujících nebo má předmět jinou váhu (např. volitelný vs povinný) a vyučované technologie a hloubka učiva se může lišit.



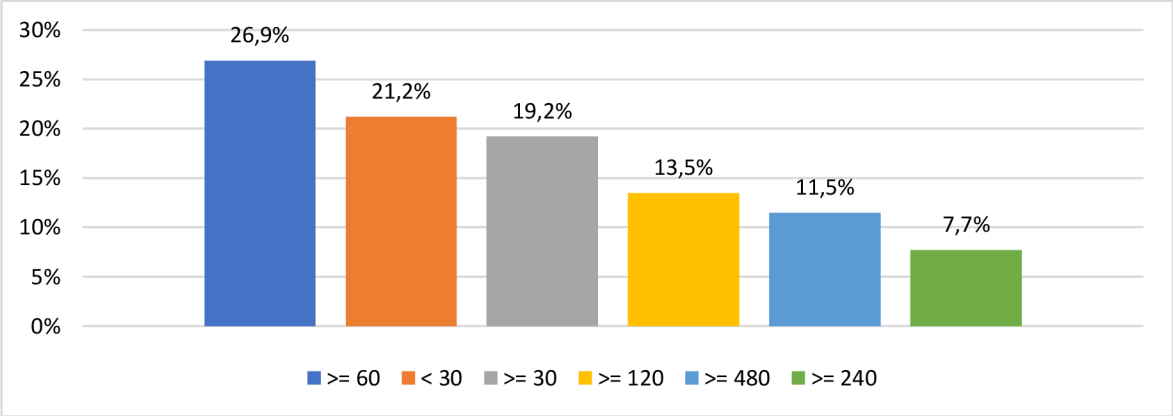
Obrázek 42: Má škola nějaké zaměření? (100 % = 52 odpovědí)

V dotazníku se vyskytlo 16 ZŠ bez zaměření, 15 SŠ odborných nebo průmyslových, 10 SŠ zaměřených na ICT, Grafiku či sítě, 4 tvoří SŠ obchodní akademie, 3 jsou SŠ zaměřené na ekonomiku a podnikání, 2 podíl tvoří Gymnázia a po 1 se vyskytly SŠ stavební a ZŠ se zaměřením na sport či jazyky.



Obrázek 43: Jaký máte počet žáků na škole (stačí odhad)? (100 % = 52 odpovědí)

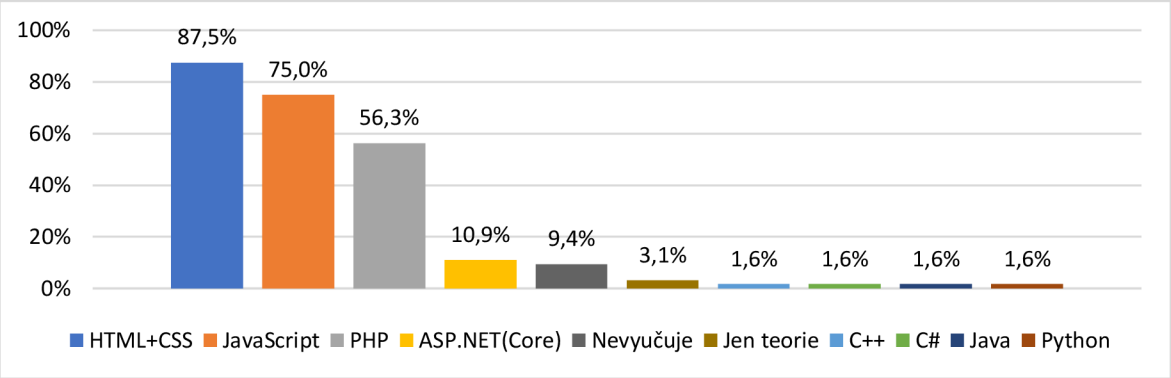
Počet uváděných žáků na škole je ve 26 případech větší než 500 a v 11 větší než 300. V 10 případech mají školy větší počet žáků než 200 a v dalších 10 větší než 100. 8 škol uvedlo počet studentů vyšší než 400 a pouze v jedné škole menší než 100.



Obrázek 44: Kolik hodin výuky webových technologií mají studenti celkem během studia? (100 % = 52 odpovědí)

Ve 14 školách mají studenti během studia 60 až 120 hodin webových technologií, což zhruba odpovídá 1–2 hodinám týdně při výuce koncentrované do jednoho roku. V 11 případech mají méně než 30 hodin. V 10 školách je rozsah výuky webových technologií v rozsahu 30 až 60 hodin za studium. V 7 mají studenti výuku v rozsahu od 120 až do 240 hodin za studium. V 6 školách je počet hodin výuky webových technologií 480 hodin a více. Ve 4 je rozsah výuky v rozsahu od 240 do 480 hodin. Průměrný počet hodin výuky webových technologií za týden se na školách pohybuje okolo 1,8 hodin a průměrný počet pololetí, během kterých žáci webové technologie studují je 2,2. Minimální počet pololetí strávený výukou je 0,1 a maximální 8. Nejčastěji se webové aplikace vyučují po dobu 2 pololetí.

13.1 Jazyky pro vývoj webu

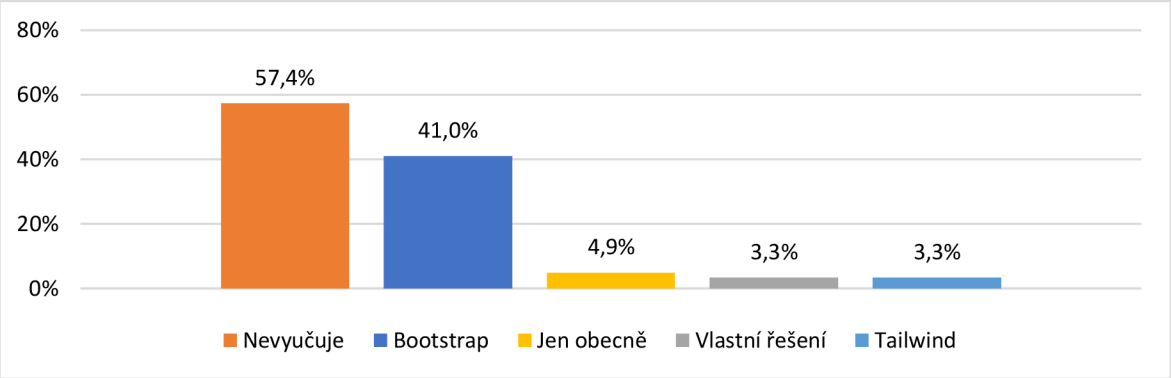


Obrázek 45: Jaké jazyky pro tvorbu webu vyučujete? (100 % = 64 odpovědí)

Vyučované technologie na školách se liší dle typu školy a typu předmětu. Na základních školách se vyučuje HTML a CSS. Je to pravděpodobně proto, že školy tyto technologie vyučují na kroužcích, případně jim věnují pár hodin v běžné výuce ICT. Nemají na to tolik času, sami učitelé nemají tolik znalostí a na studenty na základní škole i základy HTML a CSS mohou být složité. Na středních školách v předmětech, které tvoří úvod do vývoje webových stránek se vyučuje HTML, CSS a JavaScript. Na tyto technologie navazují v dalších ročnících (technicky zaměřené nebo ICT školy) nebo volitelných předmětech (technicky nezaměřené školy např. ekonomické, zemědělské) jazyky jako je PHP, C#, ASP.NET a databázové systémy či webhostingy. Dá se tedy předpokládat, že učitelé studenty nejdříve seznámí se základy webových technologií a následně se snaží propracovat až ke spuštění dynamické webové stránky na webhostingu.

V některých odpovědích z dotazníků bylo uvedeno, že školy vyučují weby pomocí CMS systémů jako je WordPress či vlastní CMS systém a další technologie neuvedly. Tedy je možné, že nemalé procento škol k výuce používá primárně (nebo i výhradně) CMS systémy a jazyky pro tvorbu webu neřeší.

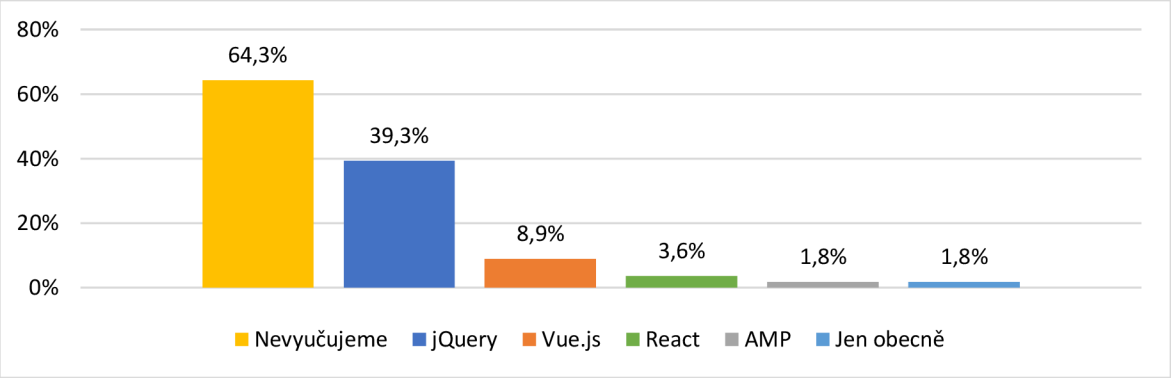
13.2 CSS frameworky



Obrázek 46: Jaké CSS frameworky vaše škola vyučujete? (100 % = 61 odpovědí)

Co je z grafu patrné, tak na školách se vyučuje Bootstrap skoro v polovině případů. Tento framework je i jedním z nejvíce požadovaných frameworků v pracovních inzerátech (viz kapitola 12.3). Tailwind, který je požadovaný ve 3 % inzerátů (viz kapitola 12.6), školy zmínily ve 3,3 %. Samostatně byl pak uveden framework Bulma.io.

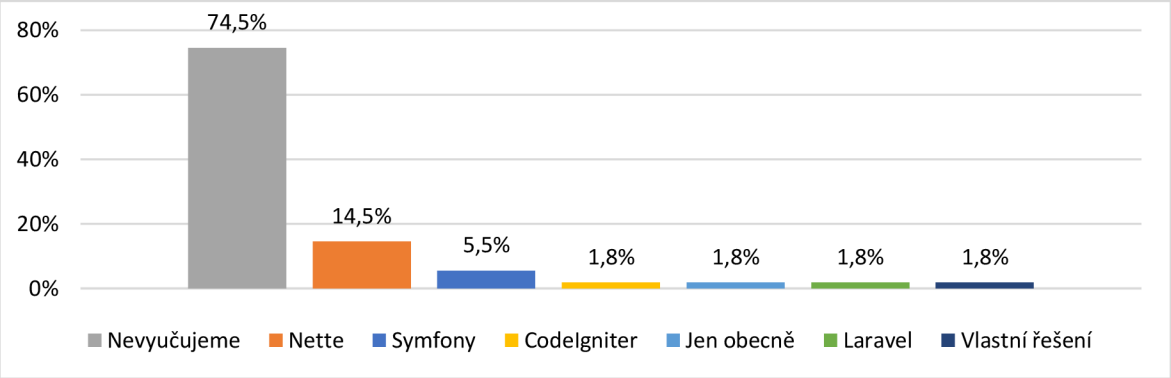
13.3 JavaScriptové frameworky



Obrázek 47: Jaké JavaScriptové frameworky vyučujete? (100 % = 56 odpovědí)

Ve více než půlce případů se na školách frameworky nevyučují. Dá se tedy usuzovat, že školy vyučují spíše čistý JavaScript, anebo ho nevyučují vůbec. Ve více než jedné třetině případů se vyučuje framework jQuery. I když tento framework již není tak žádaný na pracovním trhu (viz kap. 12.4), tak si svou pozici na školách stále udržuje. Pravděpodobně je to pro to, že učitelé ještě na nové trendy nezareagovali, nemají čas se je sami řádně naučit a do škol nikdo znalý těchto technologií nepřichází. Je možné, že jim taky nepřijde důležité a vhodné při například menší hodinové dotaci vyučovat novinky jako je Vue.js či React. Jedna škola ale uvedla, že vyučuje technologii AMP.

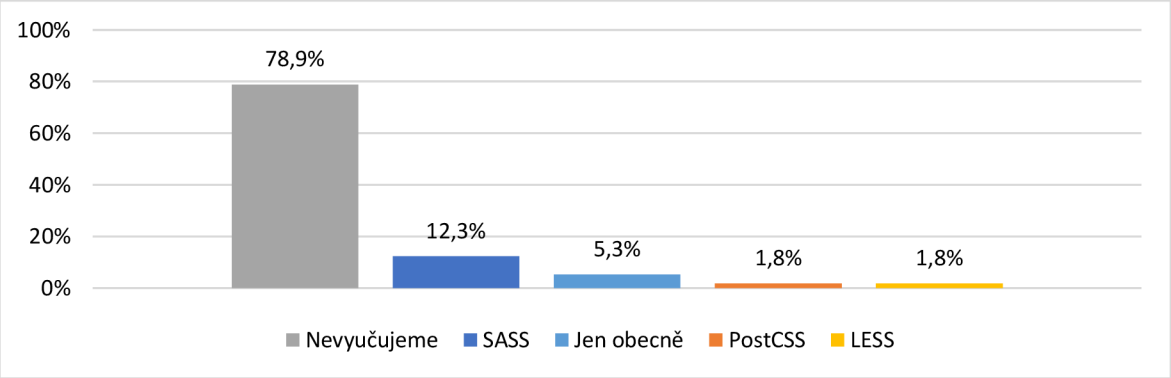
13.4 PHP frameworky



Obrázek 48: Jaké PHP frameworky vyučujete? (100 % = 57 odpovědí)

Ve skoro třech čtvrtinách případů se na školách PHP frameworky nevyučují. Někteří učitelé uvedli, že je to zbytečné a spíše se věnují funkcionalitě a teorii nebo vyučují vlastní řešení. V odpovědích byl nejčastěji uváděn Nette Framework. Důvodem, proč je zmíněn více, než například uváděná Symfony může být dokumentace a návody v češtině, česká komunita, to že je celkem žádaný na českém trhu (viz kapitola 12.5) anebo, že v něm sám vyučující pracoval a umí ho nejlépe.

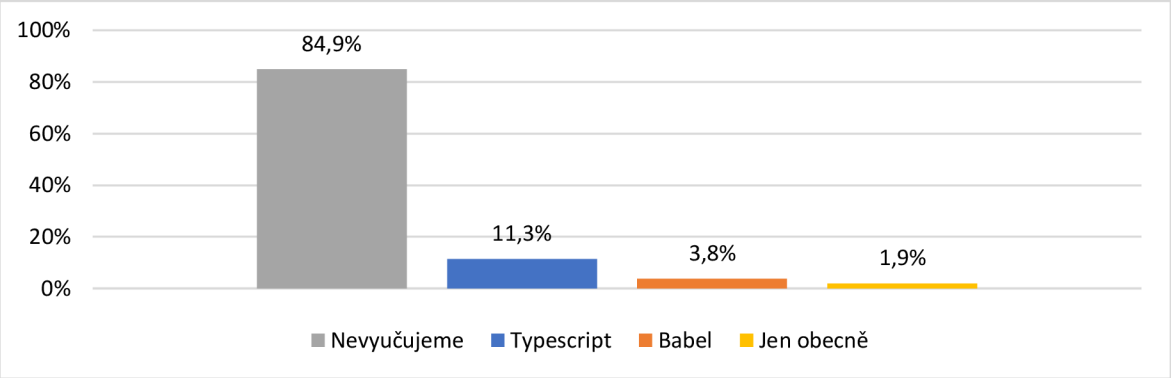
13.5 CSS preprocesory a postprocesory



Obrázek 49: Jaké CSS preprocesory a postprocesory vyučujete? (100 % = 57 odpovědí)

CSS preprocesory a postprocesory se na školách téměř nevyučují. Výjimečně se vyučuje SASS, PostCSS a LESS. Důvodem může být, že tyto technologie vyučující neznají, nevidí v nich užitek v rámci výuky a ani důležitost je vyučovat. Některé školy uvedly, že žáky o těchto technologiích informují ale vyučují čistě CSS anebo jejich výuku plánují.

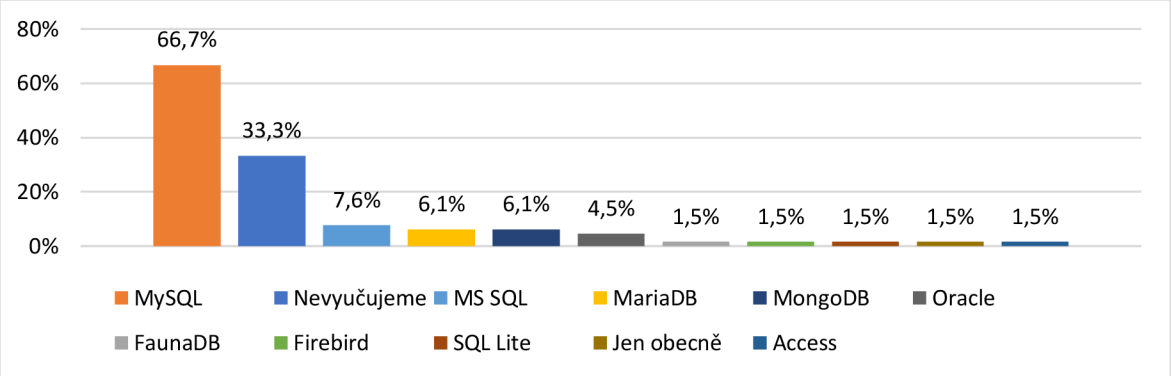
13.6 Rozšíření a kompilátory jazyka JavaScript



Obrázek 50: Jaká rozšíření a kompilátory jazyka JavaScript vyučujete? (100 % = 53 odpovědí)

Obdobně jako u CSS preprocesoru a postprocesorů výše se JavaScriptová rozšíření téměř nevyučují. V několika školách se vyučuje Typescript a Babel a například Flow, který byl v pracovních inzerátech, zde nebyl zmíněn ani jednou. Důvodem opět může být, že učitelé tyto technologie neznají případně nemají ani prostor na to je vyučovat.

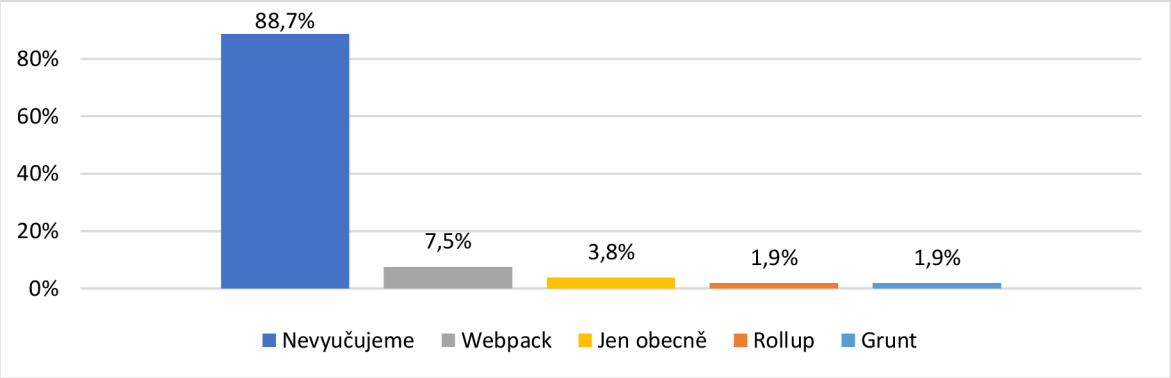
13.7 Databázové systémy



Obrázek 51: Jaké databázové systémy vyučujete? (100 % = 66 odpovědí)

Důvodem, proč má MySQL tak vysoký podíl ve výuce může být ten, že se jednoduše nainstaluje do počítače pomocí XAMPP či MAMP (viz 13.10). Naopak důvodem, proč se databáze nevyučují může být, že vyučující nemají čas je do výuky zařadit, neumí je nebo vyučují pouze statické stránky. Jelikož některé školy uvedly, že vyučují pouze CMS systémy, tak je možné, že databáze vůbec do výuky nezařazují. Z grafu je také viditelné, že výuce převažují relační databáze oproti NoSQL databázím. Což opět může být tím, že jsou vyučující zvyklí pracovat spíše s relačními databázemi a asi se i očekává, že znalost relačních databází studentům přinese větší užitek při hledání práce.

13.8 Balíčkovací nástroje

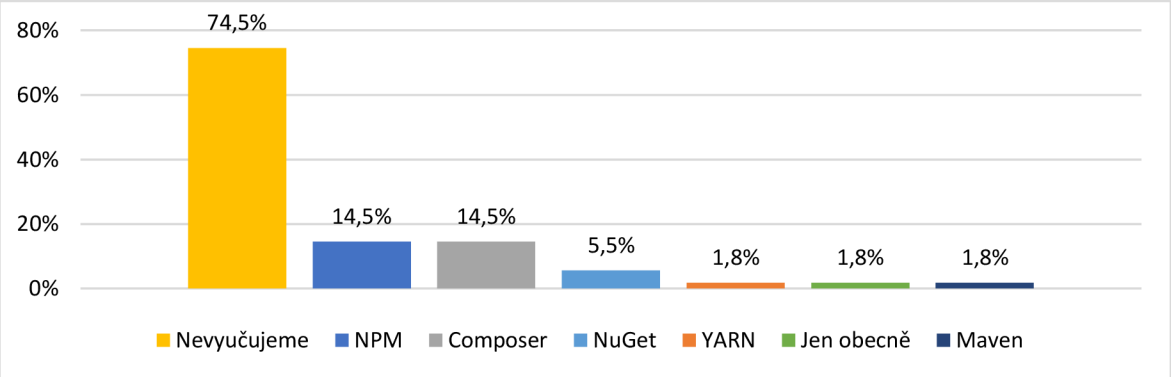


Obrázek 52: Jaké balíčkovací nástroje vyučujete? (100 % = 53 odpovědí)

U balíčkovacích nástrojů se na několika školách vyučuje Webpack, Rollup a Grunt. Většina škol tyto technologie nevyučuje.

Tyto nástroje se používají převážně u větších a komplikovanějších projektů, které mají např. i více JS a CSS souborů a už se musí dbát na optimalizaci a některé soubory je potřeba kompilovat. Je tedy možné, že z důvodu práce s malým počtem souborů školám nepřijde důležité tyto technologie vyučovat.

13.9 Nástroje pro správu a instalaci balíčků

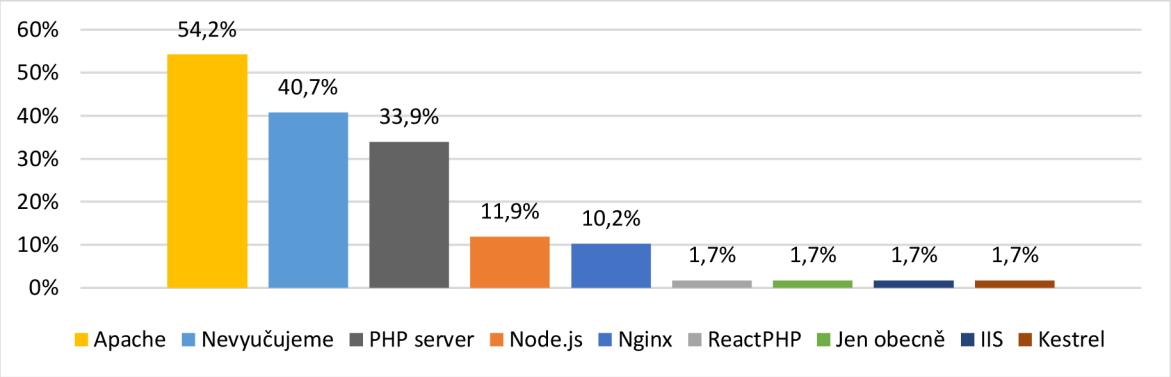


Obrázek 53: Jaké nástroje pro správu a instalaci balíčků vyučujete? (100 % = 55 odpovědí)

Jelikož školy tolik nevyučují frameworky, balíčkovací nástroje ani například kompilátory a rozšíření JavaScriptu, tak nemají moc důvod vyučovat nástroje pro správu a instalaci balíčků. Pokud ale vyučující vyučuje například PHP či JavaScriptový framework, tak zároveň uvedl, že používá NPM (v jednom případě YARN) či Composer. Důvodem, proč se ve většině případů tyto nástroje nevyučují může být, že např. pro

JavaScript studenti vše linkují přímo do HTML souborů nebo si skripty stáhnou ručně. Pro PHP a .NET mohou také použít ruční stáhnutí balíčků nebo je mohou stáhnout pomocí IDE nástrojů. Také je možné, že školy nepracují se závislostmi v takovém rozsahu, aby se jim vyplatilo tyto technologie vyučovat.

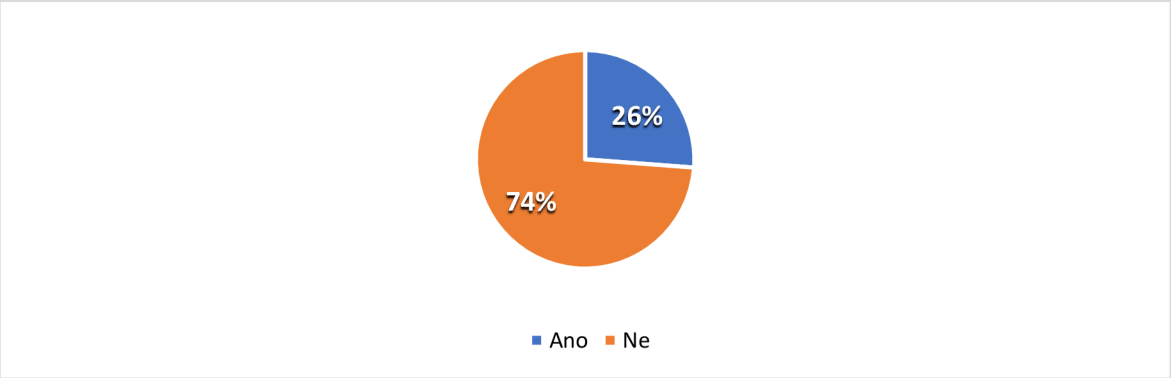
13.10 HTTP servery



Obrázek 54: Jaké vyučujete HTTP servery? (100 % = 59 odpovědí)

Dle výsledků uvedených v obrázku 54 vyučující nejčastěji uvedli používání HTTP serveru Apache. Důvodem, proč je uváděn častěji, než ostatní servery může být, že se snadno nainstalují pomocí aplikací jako je např. XAMPP (viz apachefriends.org/index.html), WAMP (viz wampserver.com/en) a MAMP (viz mamp.info/en/windows). Totéž platí pro Nginx. Dalším používaným serverem je PHP server. Tento server je také oblíben pravděpodobně z důvodu jednoduchosti používání a instalace (viz kapitola 7.3). Dalšími používanými HTTP servery jsou Node.js a ReactPHP, IIS a Kestrel. Skoro v polovině případů školy HTTP servery nevyužívají. Pokud budeme vycházet z toho, že k používání HTTP serveru školy využijí např. zmíněné nástroje XAMPP nebo IIS Express, můžeme předpokládat, že HTTP servery jako takové nejsou nijak do hloubky vyučovány a studenti jsou pouze poučeni o tom, jak je používat. Proto je zde uvedeno procento používání namísto výuky. Zároveň, pokud vyučující vyučuje pouze CMS systémy dostupné online, nemusí si u sebe studenti žádný HTTP server spouštět ani se o něm učit.

13.11 Vyučuje škola nástroj Git

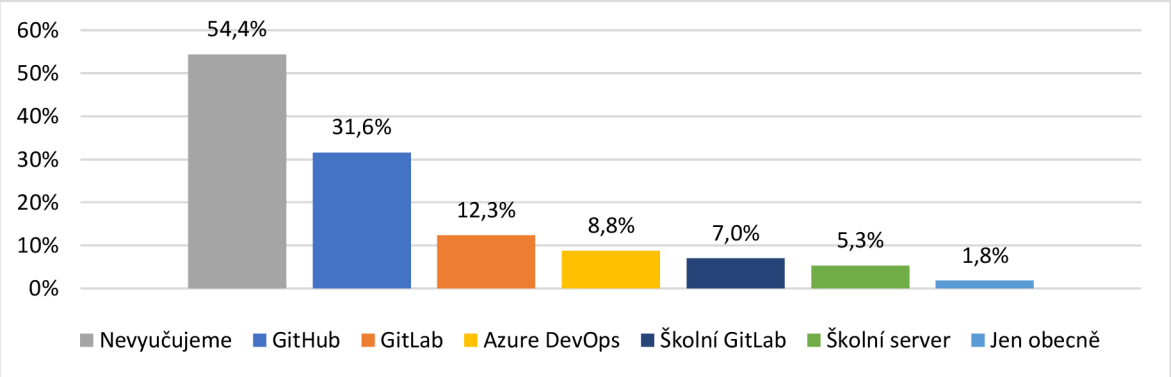


Obrázek 55: Vyučujete GIT? (100 % = 61 odpovědí)

Git je technologie, která slouží pro verzování kódu (viz kapitola 9). Tuto technologii vyučuje 26 % vyučujících a v 74 % případech se nevyučuje.

Důvodem může být, že nevidí důvod tuto technologii navíc vyučovat ani začleňovat do dalších předmětů, protože by je mohla zdržovat. Je také možné, že používají vestavěné nástroje pro práci s GITem přímo v editorech, a tedy se studenti pro základní práci příkazy učit nemusí.

13.12 Prostory pro uložení projektu

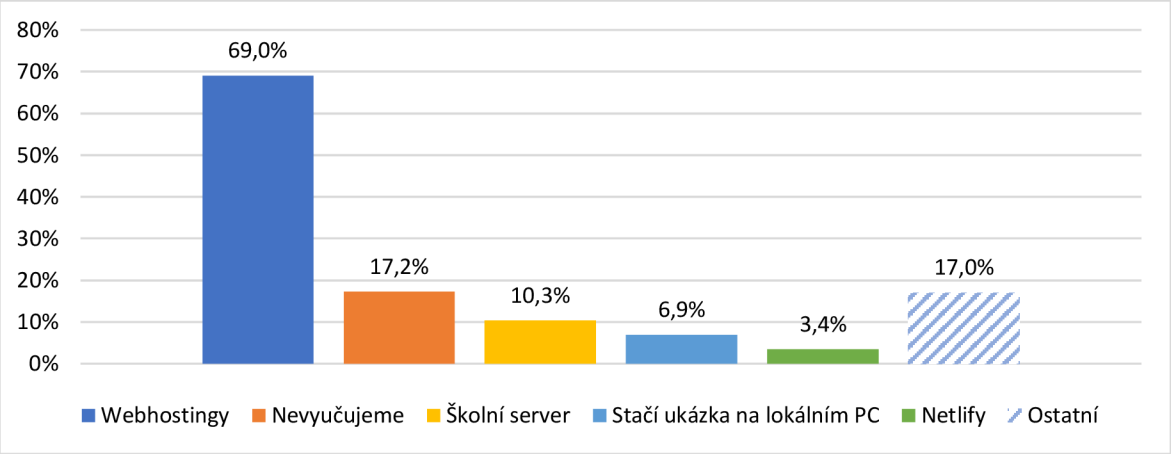


Obrázek 56: Kde mají žáci za úkol spustit školní projekt nebo práci v hodině? (100 % = 57 odpovědí)

Ve výsledcích dotazníků školy jako nejčastěji používané uložiště uvedli GitHub a GitLab následovaný Azure DevOps. Některé školy uvedly, že GitLab používají i vlastní. Pokud bychom sečetli obě možnosti používání GitLabu, tak nám dají dohromady 19,3 %. Oproti GitHubu je to ale stále skoro o polovinu méně.

Velký podíl tvoří školy, které ukládání projektu vůbec neřeší. Spadají sem jak základní, tak ale i střední technické školy. Vyučující většinou uvedli, že studenty o těchto technologiích informují, ale nevyučují je. Někteří zmínili, že se alespoň snaží ukládat kód z hodiny na GitHub pro inspiraci a aby si je tam odsud žáci stáhli a tím se s prostředím alespoň trochu naučili sami.

13.13 Prostory pro spuštění webové aplikace

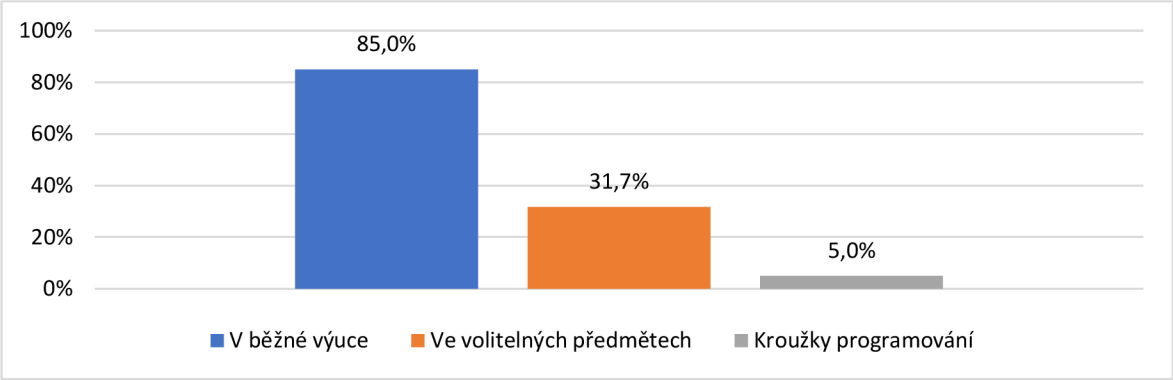


Obrázek 57: O jakých prostorách pro spuštění projektu žáky informujete nebo učíte? (100 % = 66 odpovědí)

Školy uvedly, že k nasazení projektů používají Webhostingy, a to zejména neplacené případně nasmlouvané. Další možností pro žáky, jak spustit aplikaci, je na školním serveru. Při výuce naprostých základů stačí, když žáci spustí svůj projekt nebo práci v hodině pouze u sebe na počítači, a to třeba i jako statickou stránku. Některé školy nasazení neřeší a stačí jim odevzdat soubory projektu nebo případně zaslání odkazu na web postavený na vyučovaném CMS systému. Projekty se většinou nenasazují na základních školách. Zde stačí ukázka na svém počítači.

Do sloupce ostatní spadá 7 technologií, které školy uvedly individuálně a všechny měly po 1,7 %. Mezi tyto technologie patří Vertigo, vlastní cloud, Repl.it, CodePen, Google Cloud, Microsoft Azure a Vercel. Vše zmíněno unikátně.

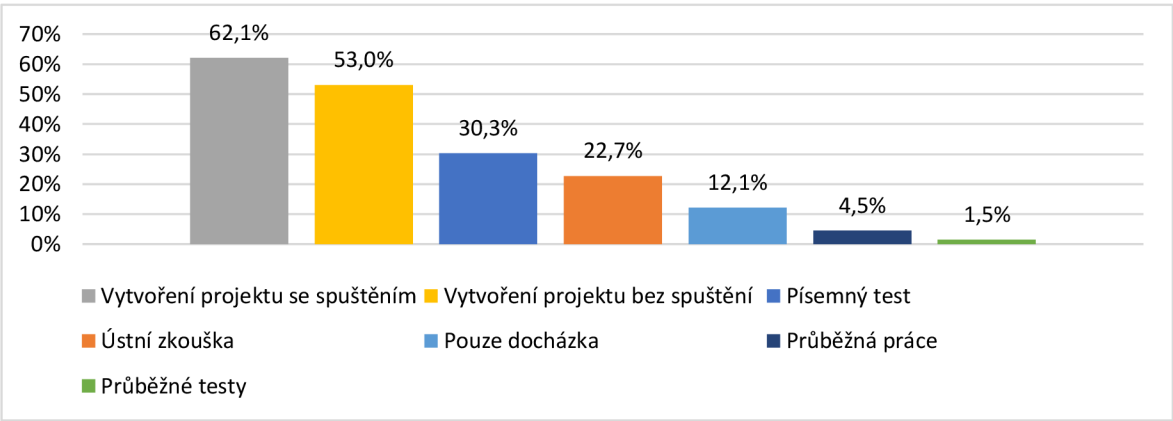
13.14 Kde a při jakých předmětech výše zmíněné technologie vyučují



Obrázek 58: Kde a při jakých předmětech výše zmíněné technologie vyučujete? (100 % = 60 odpovědí)

Webové technologie se dle výsledků z dotazníků vyučují primárně v běžné výuce. Základní školy uvedly, že pro výuku webovek mají dobrovolné kroužky. Na středních školách nezaměřených na techniku a ICT se vyučují například statické webové stránky v běžné výuce, a pokud se studenti chtějí dozvědět více, mohou si zapsat volitelný předmět, kde se učí například o databázích. Na školách zaměřených na ICT či techniku se vyučují nejdříve jeden rok statické stránky a druhý dynamické. Případně je látka postupně rozdělena do celého studia.

13.15 Co obsahuje výstupní zkouška



Obrázek 59: Co obsahuje výstupní zkouška z předmětu výuky webových technologií? (100 % = 66 odpovědí)

Vytvoření webu se spuštěním (nasazením webu na hosting) a bez spuštění (odevzdání kódu či zobrazení statické stránky) jsou výstupní požadavky ve více než polovině

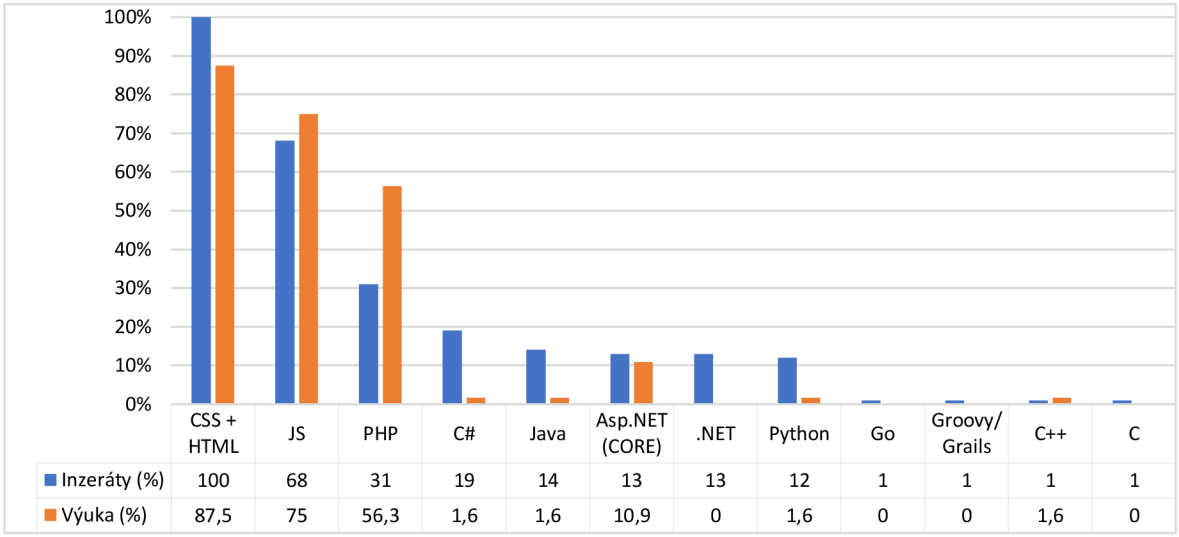
případů. Tyto případy jsou rozděleny dle toho, jestli se žáci učí o HTTP serverech, hostingových službách a například PHP či nikoliv. Pokud se vyučují pouze statické stránky, tak se odevzdává kód nebo stačí stránku odprezentovat třídě. Pokud jsou zde databáze a nějaký dynamický kód napsaný např. v PHP, tak se projekt například spouští na webhostingu. Dalšími volenými způsoby zakončení byl písemný test kombinovaný s ústní zkouškou. Pokud předmět nebyl tak důležitý (např. byl volitelný nebo jen obecný), tak učitelé volili hodnocení za docházku spojeného s odevzdáváním průběžných prací. Někteří vyučující jako formu hodnocení používají průběžné testy. Na základních školách, školách nezaměřených na techniku či ve volitelných předmětech se pak využívá i forma hodnocení ve smyslu, kam se žák za dobu výuky posunul.

13.16 Další vyučované technologie a nástroje

Kromě CMS a frameworků vlastní výroby školy k výuce využívají i aplikace Scratch (viz scratch.mit.edu) a Hour Of Code (viz hourofcode.com/cz) pro obecný a zábavný úvod do logiky a programování. Další vyučované technologie na školách jsou v Příloze 3.

14 Porovnání vyučovaných a používaných technologií ve školách oproti požadavkům na trhu práce

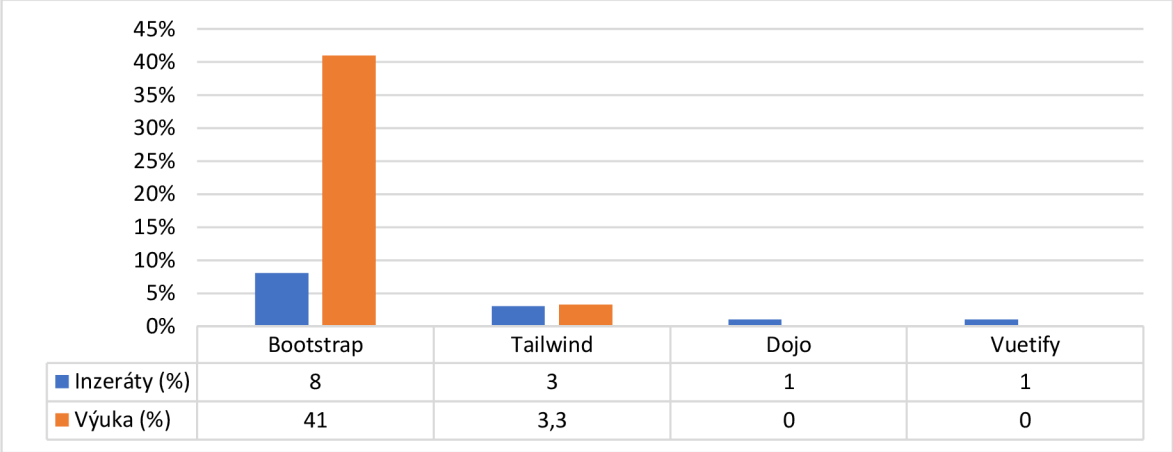
V tabulkách jsou uvedené technologie požadované v pracovních inzerátech, následně sloupec v kolika procentech inzerátů byla technologie zmíněna a v posledním sloupci v kolika procentech škol se tato technologie vyučuje. Technologie jsou vždy seřazeny od nejvíce k nejméně zmíněné v inzerátech.



	CSS + HTML	JS	PHP	C#	Java	Asp.NET (CORE)	.NET	Python	Go	Groovy/ Grails	C++	C
Inzeráty (%)	100	68	31	19	14	13	13	12	1	1	1	1
Výuka (%)	87,5	75	56,3	1,6	1,6	10,9	0	1,6	0	0	1,6	0

Obrázek 60: Porovnání jazyků pro vývoj webu (JS = JavaScript)

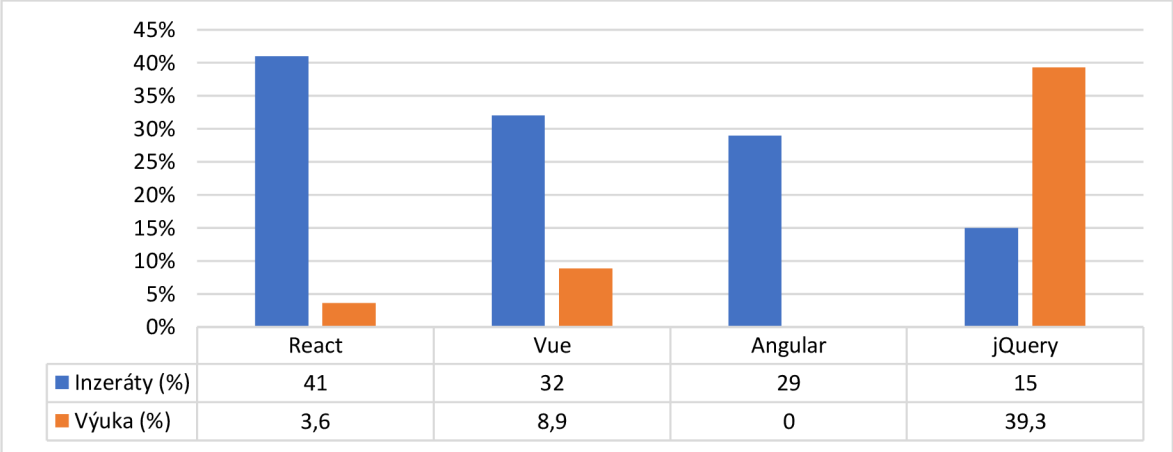
Z tabulky výše je viditelné, že první 3 požadované technologie z inzerátů se vyučují i na školách. Naopak C#, C++, C, Java na školách nejsou vyučovány tak často a Groovy/Grails a Go nejsou vyučovány vůbec. Ač se jazyk .NET nevyučuje, tak jeho jednotlivé technologie ASP.NET i C# ano. Důvodem, proč školy vyučují PHP více než bylo požadováno v inzerátech může být, že je to již zaběhlý způsob výuky tvorby webových aplikací a pomocí XAMPP či MAMP ho dokáže zprovoznit každý.



	Bootstrap	Tailwind	Dojo	Vuetify
Inzeráty (%)	8	3	1	1
Výuka (%)	41	3,3	0	0

Obrázek 61: Porovnání CSS frameworků

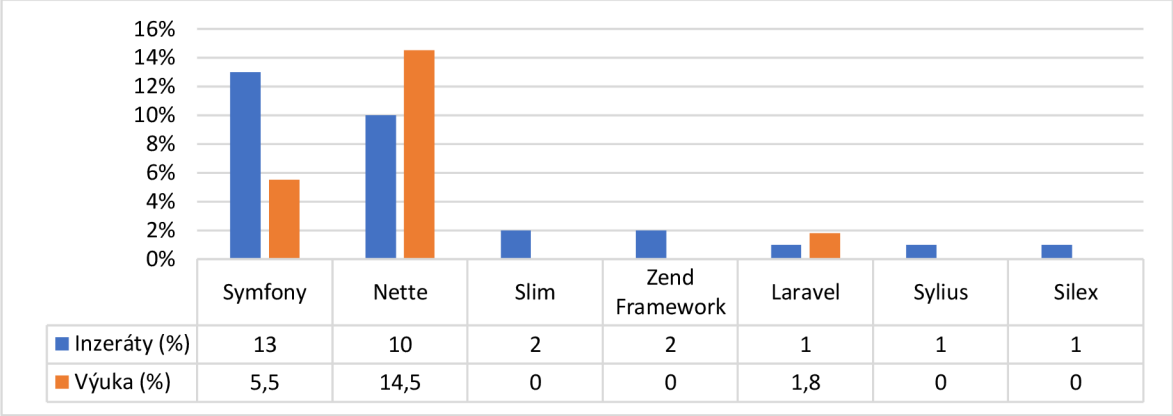
Dle tabulky výše se na školách vyučuje pracovním trhem požadovaný Bootstrap. Tailwind pouze zřídka a Dojo či Vuetify vůbec.



	React	Vue	Angular	jQuery
Inzeráty (%)	41	32	29	15
Výuka (%)	3,6	8,9	0	39,3

Obrázek 62: Porovnání JavaScriptových frameworků

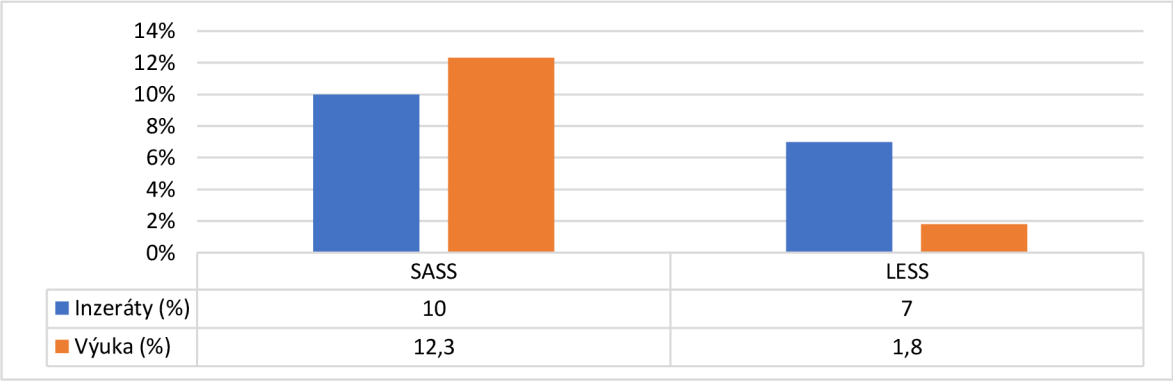
Při srovnání JavaScriptových frameworků můžeme vidět, že na školách se nejvíce vyučuje jQuery, která na pracovním trhu již není tak žádaná. Školy občas vyučují Vue.js či React. Angular se na školách nevyučuje.



	Symfony	Nette	Slim	Zend Framework	Laravel	Sylius	Silex
Inzeráty (%)	13	10	2	2	1	1	1
Výuka (%)	5,5	14,5	0	0	1,8	0	0

Obrázek 63: Porovnání PHP frameworků

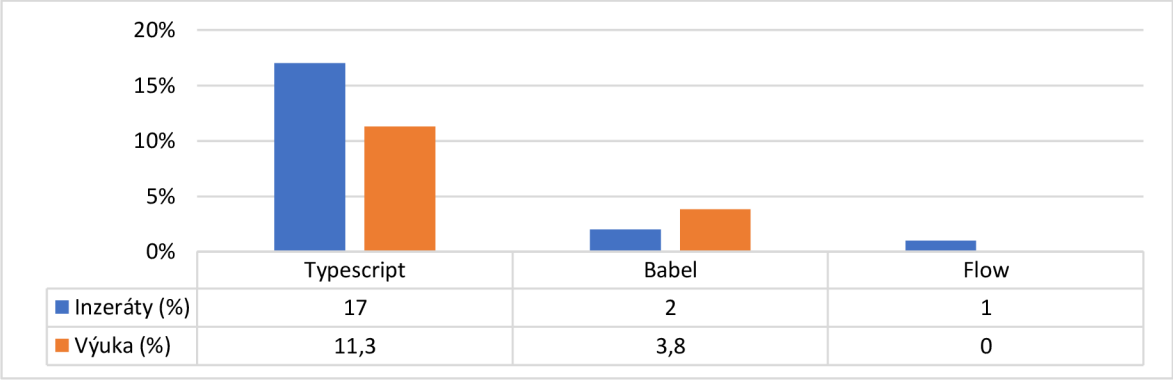
V případě PHP frameworků školy nejčastěji vyučují Nette a o trochu méně častěji Symfony. Laravel vyučuje pouze několik škol a Zend Framework, Sylius a Silex ani jedna.



	SASS	LESS
Inzeráty (%)	10	7
Výuka (%)	12,3	1,8

Obrázek 64: Porovnání CSS preprocesorů a postprocesorů

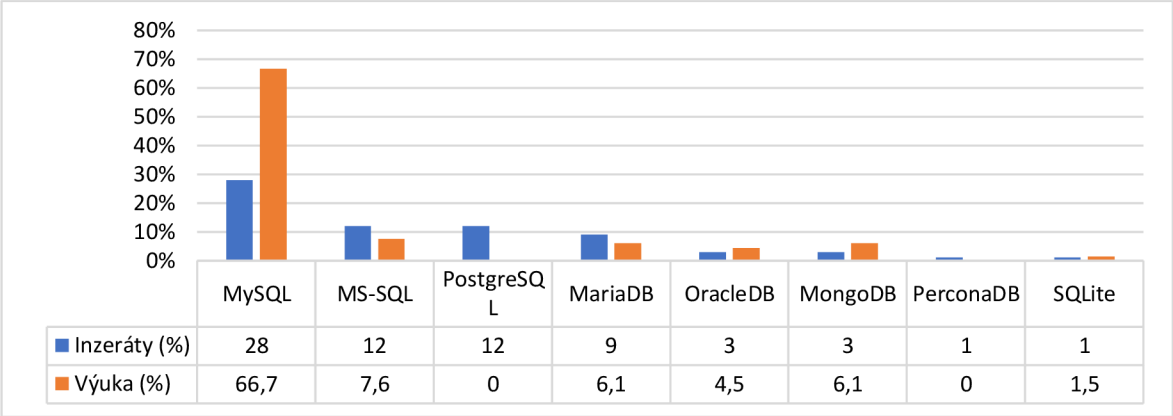
Co se týče preprocesorů a postprocesorů, tak požadovaný SASS preprocesor školy občas vyučují. Zmíněný LESS ale velmi zřídka.



	Typescript	Babel	Flow
Inzeráty (%)	17	2	1
Výuka (%)	11,3	3,8	0

Obrázek 65: Porovnání rozšíření a kompilátorů jazyka JavaScript

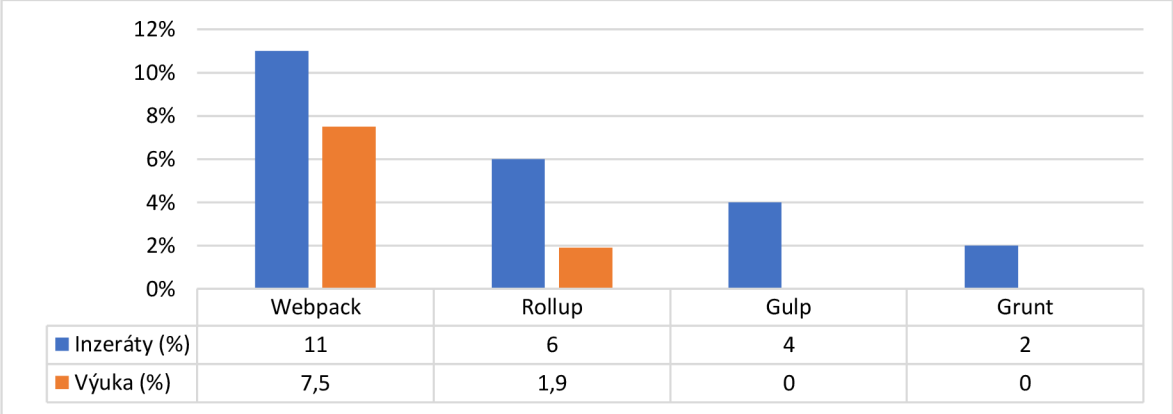
V oblasti kompilátorů je nejvíce vyučovaným nástrojem Typescript, který je i zároveň požadovaný na trhu práce. Babel školy vyučují pouze některé a Flow žádné.



	MySQL	MS-SQL	PostgreSQL	MariaDB	OracleDB	MongoDB	PerconaDB	SQLite
Inzeráty (%)	28	12	12	9	3	3	1	1
Výuka (%)	66,7	7,6	0	6,1	4,5	6,1	0	1,5

Obrázek 66: Porovnání databázových systémů

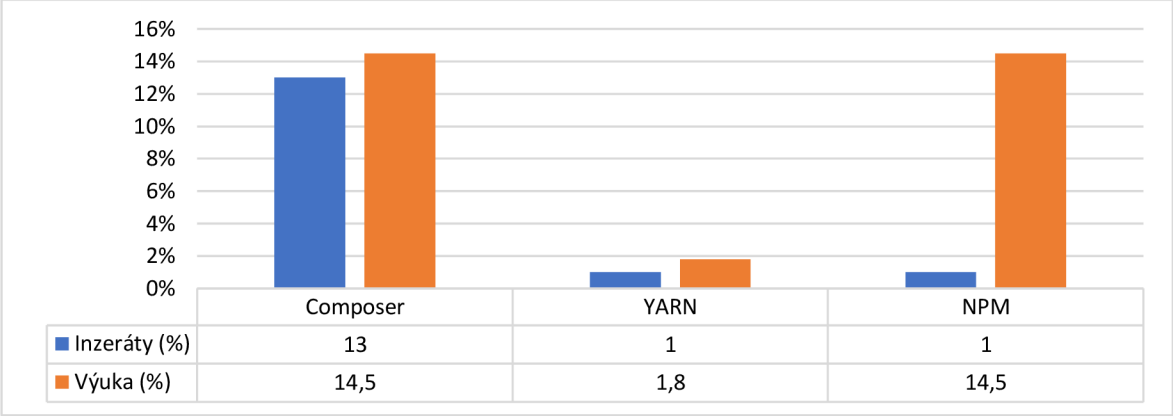
U databází můžeme říct, že požadovaná MySQL se ve školách vyučuje. Ostatní databáze ale už ne tolik a PerconaDB například vůbec.



	Webpack	Rollup	Gulp	Grunt
Inzeráty (%)	11	6	4	2
Výuka (%)	7,5	1,9	0	0

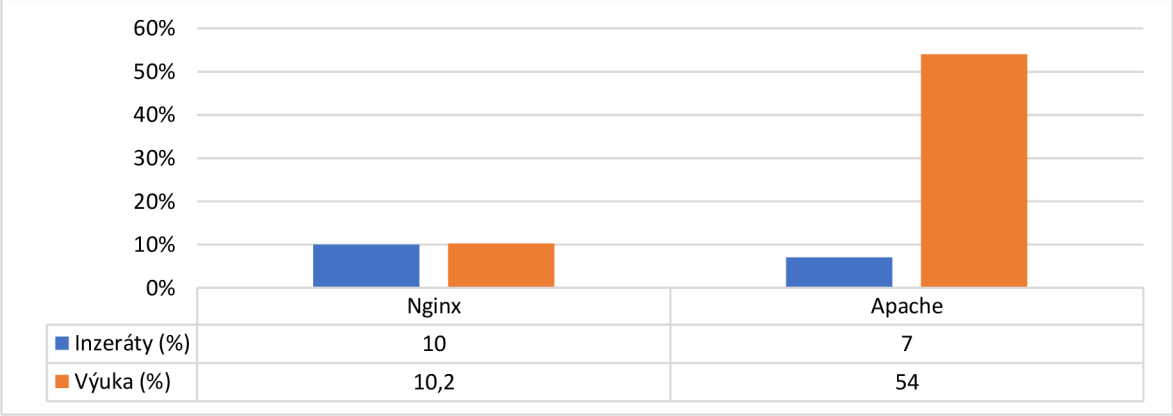
Obrázek 67: Porovnání balíčkovacích nástrojů

Požadované balíčkovací systémy se na školách vyučují jen velmi zřídka. Nejčastěji se zde vyučuje Webpack. Zmíněný Gulp nebyl v odpovědích z dotazníků ani jednou.



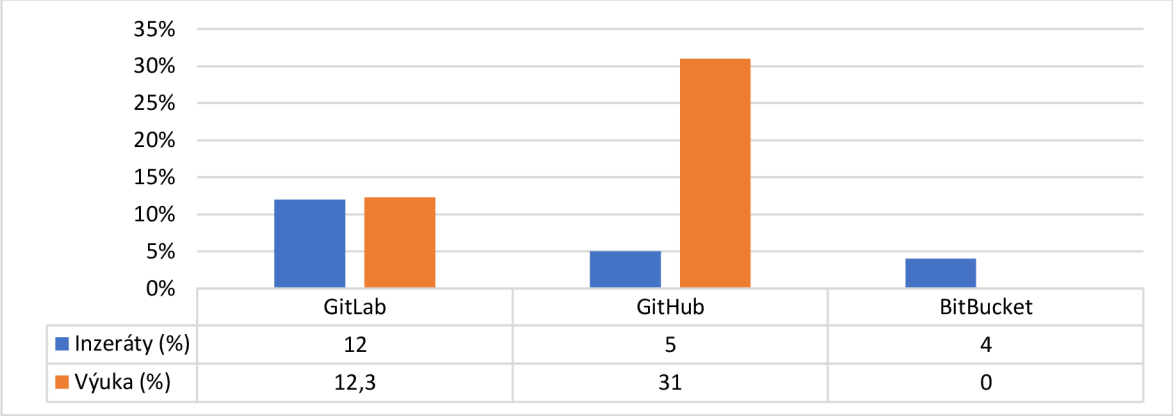
Obrázek 68: Porovnání nástrojů pro správu a instalaci balíčků

Nejvyžadovanější nástroj pro správu balíčků Composer se dle tabulky výše vyučuje i na školách. Yarn a NPM velmi málo. I když se procenta u např. YARNU téměř shodují, tak velmi pravděpodobně ve firmách tyto nástroje používají častěji. Tedy pravděpodobně poměr požadavek na trhu práce oproti výuce je horší.



Obrázek 69: Porovnání HTTP Serverů

U HTTP serverů školy nejčastěji vyučují Apache, který nebyl v inzerátech tak často. Zmíněný NGINX je vyučovaný o dost méně. Naopak na školách vyučují například Node.js a PHP server.



Obrázek 70: Porovnání prostorů pro uložení projektu

Obdobně jako v pracovních inzerátech školy nejčastěji používají a vyučují nástroj GitLab a GitHub. BitBucket ani v jednom případě.

Z výzkumu vyplývají i případné vhodné kombinace nástrojů a technologií pro vývoj webu, které by dle výzkumu mohly studentům zvýšit šanci za získání pracovní pozice. Níže jsou uvedené kombinace pro PHP a .NET:

- **Pro PHP**
  - **PHP frameworky (viz kap. 3.3):** Symfony nebo Nette
  - **Databázový systémy (viz kap. 8):** MySQL nebo PostgreSQL
  - **JavaScriptové frameworky (viz kap. 3.2):** Vue nebo React
  - **CSS frameworky (viz kap. 3.1):** Bootstrap nebo Tailwind
  - **HTTP servery (viz kap. 7):** Apache nebo Nginx pomocí XAMPP či MAMP
  - **Ostatní:** Composer (viz kap. 2.2.1), NPM (viz kap. 2.1.1), Webpack (viz kap. 6.1), GitHub (viz kap. 10.1) a Typescript (viz kap. 5.2)
- **Pro .NET**
  - **Databázový systémy (viz kap. 8):** MySQL nebo MS-SQL
  - **JavaScriptové frameworky (viz kap. 3.2):** Vue nebo React
  - **CSS Frameworky (viz kap. 3.1):** Boostrap nebo Tailwind
  - **HTTP servery (viz kap. 7):** IIS nebo IIS Express pomocí Visual Studia
  - **Ostatní:** NuGet (viz kap. 2.3.1), NPM (viz kap. 2.1.1), Webpack (viz kap. 6.1), GitHub (viz kap. 10.1) a Typescript (viz kap. 5.2)

15 Závěr

Diplomová práce se zabývá komparací technologií pro výuku tvorby webových stránek. V této práci byly technologie nejdříve představeny. Následně byly okomentovány jednotlivé sekce v oblasti statistik požadovaných znalostí na uchazeče v pracovních inzerátech. Jako poslední byly okomentovány sekce s vyučovanými technologiemi v oblasti tvorby webových stránek na základních a středních školách.

Představení jednotlivých technologií obsahovalo stručnou historii, informace o autorech a případně ukázku kódu nebo snímku z administrace, případně vysvětlení, jak a kde se nástroj používá. Zároveň byly uvedeny výhody a nevýhody jednotlivých nástrojů, které také byly vzájemně porovnávány.

Pro zmapování používaných technologií bylo použito 100 pracovních inzerátů ze 12 různých inzerentních webů, které zahrnovaly, jak pracovní portály, tak weby různých společností. Při vyhledávání pracovnách inzerátů byla použita klíčová slova používaná pro vyhledání pracovní pozice v oblasti webových technologií.

Statistiky týkající se vyučovaných technologií na základní a středních školách byly vytvořeny z odpovědí na otázky dotazníku vytvořeného na Google Forms. Celkem bylo získáno 66 odpovědí z 52 různých škol.

V kapitole 14 je v tabulkách porovnáno celkem 52 nástrojů a technologií z hlediska procent uvedení v inzerátech vůči procentům uvedení v odpovědích z dotazníků rozeslaných do škol. Z porovnání vyplývá, že pouze 7 uvedených technologií se na školách vyučuje případně používá alespoň ve 30 %. Můžeme si také všimnout, že vyučovaná jQuery spadající do této oblasti již není na trhu práce tak žádaná, což může být důsledkem opožděnější reakce školství na současné potřeby trhu. Technologií, které se vyučují alespoň v 10 % je 9. Počet nástrojů, které jsou vyučovány v rozsahu od 0,1 do 9,9 % je 19 a 17 technologií uvedených v inzerátech se nevyučuje vůbec.

Je tedy patrné, že se na školách požadované technologie na trhu práce vyučují a používají velmi málo anebo se vyučovat teprve začínají až v době, kdy už na trhu pomalu zase mizí. Namísto zaměření se na konkrétní technologie jsou využívány vlastní řešení (např. vlastní frameworky či CMS systémy) a občas podobné technologie (např. namísto Symfony vyučují CodeIgniter). Některé vyučované technologie zároveň

již nejsou aktuální (viz vyučovaná jQuery Obrázek 62). Oblasti, které jsou na tom nejlépe jsou jazyky pro vývoj webů (viz Obrázek 60), databázové systémy (viz Obrázek 66), nástroje pro správu balíčků (viz Obrázek 68), HTTP servery (viz Obrázek 69) a prostory pro uložení projektu (viz Obrázek 70). Naopak oblasti, kde se statistiky z dotazníků nejvíce rozchází s požadavky jsou frameworky (viz Obrázek 61, Obrázek 62, Obrázek 63) a balíčkovací nástroje (viz Obrázek 67). Důvodem, proč se frameworky a balíčkovací nástroje nevyučují tak hojně je asi to, že školy se spíše zaměřují na výuku programovacích a značkovacích jazyků, práci s databázemi, obecnou instalaci knihoven a software a spuštění projektu. S tímto přístupem se studenti naučí základy a programování webů obecně. Případné knihovny a nástroje pro složitější projekty se pak naučí používat sami.

Je tedy žádoucí žáky vést spíše ke zvyku neustále se samostatně učit novým věcem a mít přehled o novinkách, případně vědět, kde se o nich dovídat a jak získávat aktuální informace. Zároveň je vhodnější si pro výuku vybrat technologii, která aktuálně na trhu práce je (a zároveň případně již dlouho před tím byla a je ověřená) a vyučovat na ni konkrétní principy a vývojářské postupy (např. OOP a DI), které jsou napříč technologiemi sdílené, namísto výuky několika různých technologií okrajově. Pokud se totiž žáci učí mnoho různých technologií okrajově, tak může nastat situace, kdy se dostanou na trh práce a daná technologie bude již zastaralá.

16 Seznam použité literatury

16.1 Odborné publikace

[op-1] VRÁNA, Jakub. *1001 tipů a triků pro PHP*. Brno: Computer Press, 2010. ISBN 978-80-251-2940-1.

[op-2] MICHÁLEK, Martin. Vzhůru do (responzivního) webdesignu. Verze 1.1. Praha: vlastním nákladem autora, 2017. ISBN 978-80-88253-00-6.

[op-3] PEHLIVANIAN, Ara a Don NGUYEN. JavaScript okamžitě. Brno: Computer Press, 2014. ISBN 978-80-251-4163-2.

[op-4] STEPHENS, Ryan K., Ronald R. PLEW a Arie JONES. Naučte se SQL za 28 dní: [stačí hodina denně]. Brno: Computer Press, 2010. ISBN 978-80-251-2700-1.

[op-5] KOFLER, Michael a Bernd ÖGGL. PHP 5 a MySQL 5: průvodce webového programátora. Brno: Computer Press, 2007. ISBN 978-80-251-1813-9.

[op-6] KENT, Jeffrey A. Visual C# 2005 - bez předchozích znalostí: průvodce pro samouky. Brno: Computer Press, 2007. ISBN 978-80-251-1584-8.

[op-7] MICHÁLEK, Martin a Robin POKORNÝ. Vzhůru do AMP [online]. Praha: Michálek Martin – Vzhůru dolů, 2019 [cit. 2021-01-03]. ISBN 978-80-88253-06-8. Dostupné z: https://www.vzhurudolu.cz/ebook-amp/info

[op-8] MAKZAN. Programujeme hry v HTML5. Brno: Computer Press, 2012. ISBN 978-80-251-3731-4.

[op-9] DUCKETT, Jon. HTML & CSS: design and build websites. Indianapolis, IN: Wiley, [2011]. ISBN 978-1-118-00818-8.

[op-10] MICHÁLEK, Martin. Vzhůru do CSS3 [online]. Praha: Michálek Martin – Vzhůru dolů, 2017 [cit. 2021-01-03]. ISBN 978-80-260-8438-9. Dostupné z: https://www.vzhurudolu.cz/ebook-css3/

[op-11] PORCELLO, Eve a Alex BANKS. Learning GraphQL: declarative data fletching for modern web apps. Sebastopol,CA: O´Reilly, 2018. ISBN 978-1-492-03071-3.

[op-12] BRAND, Wiley. APIs for dummies [online]. 3rd IBM Limited Edition. John Wiley & Sons, 2018 [cit. 2021-01-03]. ISBN 978-1-119-45715-2. Dostupné z: https://www.ibm.com/downloads/cas/GJ5QVQ7X

16.2 Webové stránky

[w-1] History of PHP. PHP.net [online]. [cit. 2021-01-03]. Dostupné z: https://www.php.net/manual/en/history.php.php

[w-2] What is PHP? PHP.net [online]. [cit. 2021-01-03]. Dostupné z: https://www.php.net/manual/en/intro-whatis.php

[w-3] Built-in web server. PHP.net [online]. [cit. 2021-01-03]. Dostupné z: https://www.php.net/manual/en/features.commandline.webserver.php

[w-4] What is ASP.NET? Microsoft.com [online]. [cit. 2021-01-03]. Dostupné z: https://dotnet.microsoft.com/learn/aspnet/what-is-aspnet

[w-5] TAMENAOUL, Hamza. Framework vs library vs package vs module: The debate. Dev.to [online]. 2020, 16. 4. 2020 [cit. 2021-01-03]. Dostupné z: https://dev.to/hamza/framework-vs-library-vs-package-vs-module-the-debate-3jpp

[w-6] A brief history of Node.js. Nodejs.dev [online]. [cit. 2021-01-03]. Dostupné z: https://nodejs.dev/learn/a-brief-history-of-nodejs

[w-7] Npm Docs. Docs.npmjs.com [online]. [cit. 2021-01-03]. Dostupné z: https://docs.npmjs.com

[w-8] NAKAZAWA, Christoph, Sebastian MCKENZIE a Jamie KYLE. Yarn: a new package manager for JavaScript. Engineering.fb.com [online]. [cit. 2021-01-03]. Dostupné z: https://engineering.fb.com/2016/10/11/web/yarn-a-new-package-manager-for-javascript

[w-9] Documentation. Getcomposer.org [online]. [cit. 2021-01-03]. Dostupné z: https://getcomposer.org/doc

[w-10] An introduction to NuGet. Docs.microsoft.com [online]. 2019-05-24 [cit. 2021-01-03]. Dostupné z: https://docs.microsoft.com/en-us/nuget/what-is-nuget

[w-11] Why should i use a framework? Symfony.com [online]. [cit. 2021-01-03]. Dostupné z: https://symfony.com/why-use-a-framework

[w-12] WATHAN, Adam. Tailwind CSS: From Side-Project Byproduct to Multi-Million Dollar Business. Adamwathan.me [online]. 2020-08-02 [cit. 2021-01-03]. Dostupné z: https://adamwathan.me/tailwindcss-from-side-project-byproduct-to-multi-mullion-dollar-business

[w-13] Utility-First. Tailwindcss.com [online]. [cit. 2021-01-03]. Dostupné z: https://tailwindcss.com/docs/utility-first

[w-14] About. Getbootstrap.com [online]. [cit. 2021-01-03]. Dostupné z: https://getbootstrap.com/docs/4.1/about/overview

[w-15] History. Jquery.org [online]. [cit. 2021-01-03]. Dostupné z: https://jquery.org/history

[w-16] JQuery. Jquery.org [online]. [cit. 2021-01-03]. Dostupné z: https://jquery.com

[w-17] Introduction. Vuejs.org [online]. [cit. 2021-01-03]. Dostupné z: https://vuejs.org/v2/guide

[w-18] What is Vuex? Vuex.vuejs.org [online]. [cit. 2021-01-03]. Dostupné z: https://vuex.vuejs.org

[w-19] Easily create user first websites with AMP. Amp.dev [online]. [cit. 2021-01-03]. Dostupné z: https://amp.dev/about/websites

[w-20] Symfony 5. Symfony.com [online]. [cit. 2021-01-03]. Dostupné z: https://symfony.com/5

[w-21] POTENCIER, Fabien. The end of the Symfony Standard Edition. Symfony.com [online]. 2018, 2018-01-10 [cit. 2021-01-03]. Dostupné z: https://symfony.com/blog/the-end-of-the-symfony-standard-edition

[w-22] Proč používat Nette? Nette.org [online]. [cit. 2021-01-03]. Dostupné z: https://doc.nette.org/cs/3.1/why-use-nette

[w-23] Componette.org. Componette.org [online]. [cit. 2021-01-03]. Dostupné z: https://componette.org

[w-24] ČÁPKA, David. Lekce 1 - Standardy jazyka PHP - Úvod a PSR-1. Itnetwork.cz [online]. [cit. 2021-01-03]. Dostupné z: https://www.itnetwork.cz/php/psr/standardy-jazyka-php-uvod-a-psr-1

[w-25] Sass Basics. Sass-lang.com [online]. [cit. 2021-01-03]. Dostupné z: https://sass-lang.com/guide

[w-26] PostCSS: a tool for transforming CSS with JavaScript. Postcss.org [online]. [cit. 2021-01-03]. Dostupné z: https://postcss.org

[w-27] What is Babel?: a tool for transforming CSS with JavaScript. Babeljs.io [online]. [cit. 2021-01-03]. Dostupné z: https://babeljs.io/docs/en

[w-28] Typed JavaScript at Any Scale. Typescriptlang.org [online]. [cit. 2021-01-03]. Dostupné z: https://www.typescriptlang.org

[w-29] Tree Shaking. Webpack.js.org [online]. [cit. 2021-01-03]. Dostupné z: https://webpack.js.org/guides/tree-shaking

[w-30] Code Splitting. Webpack.js.org [online]. [cit. 2021-01-03]. Dostupné z: https://webpack.js.org/guides/code-splitting

[w-31] Package exports. Webpack.js.org [online]. [cit. 2021-01-03]. Dostupné z: https://webpack.js.org/guides/package-exports

[w-32] Getting Started. Webpack.js.org [online]. [cit. 2021-01-03]. Dostupné z: https://webpack.js.org/guides/getting-started/#using-a-configuration

[w-33] Introduction. Rollupjs.org [online]. [cit. 2021-01-03]. Dostupné z: https://rollupjs.org/guide/en

[w-34] Parcel: Blazing fast, zero configuration web application bundler. Parceljs.org [online]. [cit. 2021-01-03]. Dostupné z: https://parceljs.org

[w-35] What is NGINX? Nginx.com [online]. [cit. 2021-01-03]. Dostupné z: https://www.nginx.com/resources/glossary/nginx

[w-36] How do i create a HTTP server? Nodejs.org [online]. [cit. 2021-01-03]. Dostupné z: https://nodejs.org/en/knowledge/HTTP/servers/how-to-create-a-HTTP-server

[w-37] Event-driven, non-blocking I/O with PHP. Reactphp.org [online]. [cit. 2021-01-03]. Dostupné z: https://reactphp.org

[w-38] Rethinking the database for a new generation of applications. Fauna.com [online]. [cit. 2021-01-03]. Dostupné z: https://fauna.com/about

[w-39] What is Fauna? Docs.fauna.com [online]. [cit. 2021-01-03]. Dostupné z: https://docs.fauna.com/fauna/current/concepts/index.html

[w-40] Getting Started - What is Git? Git-scm.com [online]. [cit. 2021-01-03]. Dostupné z: https://git-scm.com/book/en/v2/Getting-Started-What-is-Git%3F

[w-41] VFS for Git: Git at Enterprise Scale. Vfsforgit.org [online]. [cit. 2021-01-03]. Dostupné z: https://vfsforgit.org

[w-42] Where the world builds software. GitHub.com [online]. [cit. 2021-01-03]. Dostupné z: https://github.com

[w-43] GitHub Pages: Websites for you and your projects. Pages.GitHub.com [online]. [cit. 2021-01-03]. Dostupné z: https://pages.github.com

[w-44] GitLab is the open DevOps platform. About.GitLab.com [online]. [cit. 2021-01-03]. Dostupné z: https://about.giitlab.com

[w-45] Modern web apps shipped faster. Netlify.com [online]. [cit. 2021-01-03]. Dostupné z: https://www.netlify.com

[w-46] Netlify Pricing. Netlify.com [online]. [cit. 2021-01-03]. Dostupné z: https://www.netlify.com/pricing

[w-47] Develop. Preview. Ship. Vercel.com [online]. [cit. 2021-01-03]. Dostupné z: https://vercel.com

[w-48] Official Runtimes. Vercel.com [online]. [cit. 2021-01-03]. Dostupné z: https://vercel.com/docs/runtimes

[w-49] ČERNÝ, Václav. Nejlepší webhostingy – velké srovnání 2021. Testado.cz [online]. 2020, 2020-02-17 [cit. 2021-01-03]. Dostupné z: https://www.testado.cz/nejlepsi-webhostingy

[w-50] Webhosting, VPS nebo dedikovaný server? Wedos.cz [online]. [cit. 2021-01-03]. Dostupné z: https://www.wedos.cz/hosting-srovnani

[w-51] Google Cloud Pricing Calculator. Cloud.google.com [online]. [cit. 2021-01-03]. Dostupné z: https://cloud.google.com/products/calculator

[w-52] Build cross-platform desktop apps with JavaScript, HTML, and CSS [online]. [cit. 2021-01-03]. Dostupné z: https://www.electronjs.org

[w-53] MICHÁLEK, Martin. Progresivní webové aplikace: Co to je? a jak webu zařídit plné hodnocení PWA v Lighthouse [online]. 2019 [cit. 2021-01-03]. Dostupné z: https://www.vzhurudolu.cz/prirucka/pwa

[w-54] Recommended Doctype Declarations to use in your Web document. [online]. 2002 [cit. 2021-01-03]. Dostupné z: https://www.w3.org/QA/2002/04/valid-dtd-list.html

[w-55] MICHÁLEK, Martin. Jednotky pro tvorbu webu (em, rem, %, px, vh, vw): Kde použít jakou? [online]. 2018 [cit. 2021-01-03]. Dostupné z: https://www.vzhurudolu.cz/prirucka/jednotky#rem

[w-56] MACHÁČEK, Vladimír. AMP e-maily [online]. 2019 [cit. 2021-01-03]. Dostupné z: https://www.youtube.com/watch?v=uJJ6nyUwQYo&ab_channel=Frontendisti

[w-57] About issues [online]. [cit. 2021-01-03]. Dostupné z: https://docs.github.com/en/github/managing-your-work-on-github/about-issues

[w-58] Automate your workflow from idea to production [online]. [cit. 2021-01-03]. Dostupné z: https://github.com/features/actions

[w-59] Working with GitHub in VS Code [online]. [cit. 2021-01-03]. Dostupné z: https://code.visualstudio.com/docs/editor/github

[w-60] Lifecycle hooks [online]. 2021-03-02 [cit. 2021-01-03]. Dostupné z: https://v3.vuejs.org/api/options-lifecycle-hooks.html

[w-61] CDN [online]. 2020-09-15 [cit. 2021-01-03]. Dostupné z: https://developer.mozilla.org/en-US/docs/Glossary/CDN

[w-62] Render Functions & JSX: Nodes, Trees, and the Virtual DOM. Vuejs.org [online]. [cit. 2021-01-03]. Dostupné z: https://vuejs.org/v2/guide/render-function.html#Nodes-Trees-and-the-Virtual-DOM

[w-63] Virtual DOM and Internals: What is the Virtual DOM? Reactjs.org [online]. [cit. 2021-01-03]. Dostupné z: https://reactjs.org/docs/faq-internals.html

[w-64] DOM tree. Javascript.info [online]. [cit. 2021-01-03]. Dostupné z: https://javascript.info/dom-nodes

[w-65] , deathmood. How to write your own Virtual DOM. Medium.com [online]. 2016-06-01 [cit. 2021-01-03]. Dostupné z: https://medium.com/@deathmood/how-to-write-your-own-virtual-dom-ee74acc13060

[w-66] PHP on Windows. Php.iis.net [online]. 2O21n. l. [cit. 2021-01-03]. Dostupné z: https://php.iis.net/

[w-67] ANDERSON, Rick a olprod. Podpora služby IIS při vývoji v sadě Visual Studio pro ASP.NET Core. Docs.microsoft.com [online]. 2020 [cit. 2021-01-03]. Dostupné z: https://docs.microsoft.com/cs-cz/aspnet/core/host-and-deploy/iis/development-time-iis-support?view=aspnetcore-5.0

[w-68] An incredible web server that's built around you. Iis.net [online]. 2020 [cit. 2021-01-03]. Dostupné z: https://www.iis.net/overview

17 Seznam zkratek

- AJAX Asynchronous JavaScript and XML
- AMP Accelerated Mobile Pages
- API Application programming interface
- CD Continous Delivery
- CI Continuous Integration
- CLI Command Line Interface
- CSS Cascading Style Sheets
- DI Dependenci Injection
- DOM Document Object Model
- FQL Fauna Query Language
- GVFS Git Virtual File System
- HTML HyperText Markup Language
- HTTP Hypertext Transfer Protocol
- HTTPS Hypertext Transfer Prodocol Secure
- MVC Model View Controller
- MVP Model View Presenter
- NoSQL Non SQL or Non relation database
- NPM Node Package Manager
- OOP Object Oriented Programming
- PHP Hypertext Preprocesor
- PSR PHP Standard Recommendation
- SASS Syntactically Awesome Style sheets
- SEO Search Engine Optimization
- SSR Server Side Rendering
- SQL Structured Query Language
- VDOM Virtual DOM
- Verzování Vytvářet a ukládat různé verze aplikace
- VPS Virtual Private Server
- XHTML Extensible hypertext markup language

18 Seznam obrázků

Přílohy

Příloha 1: Seznam inzerentů použitých ve výzkumu

- Advantage Consulting, s.r.o.
- Aktuálně.cz (Economia, a.s.)
- Alza a.s.
- ATLAS software a.s., Avocode, Inc
- BlueGhost.cz, s.r.o.
- Chaos Czech a.s.
- CDN77.com (DataCamp s.r.o.)
- Coleman, S.I., a.s.
- Comtesys, spol. s r.o.
- CZ.NIC, z. s. p. o.
- CZC.cz s.r.o.
- Česká národní banka
- Česká potravinářská obchodní a.s.
- Česká spořitelna, a.s.
- Česká televize
- ČEZ Prodej, a. s.,
- Datamole s.r.o.
- Digiteq Automotive s.r.o.
- E LINKX a.s.
- Elektroline, a.s.
- Etimos Human s.r.o.
- ForCamping s.r.o.
- Foxconn Technology CZ s.r.o.
- Giant interactive s.r.o.
- GIT Consult Czech s.r.o.
- Hays Czech Republic s.r.o
- Heureka Shopping s.r.o.
- Historie Kermi s.r.o
- ICZ a.s.
- INIZIO Internet Media s.r.o.
- Innetic s.r.o.
- Internet Mall, a.s.
- Inveo.cz s.r.o.
- Komerční banka, a.s.
- Košík.cz s.r.o.
- Kupi.cz retail, s.r.o
- Laboratory Imaging, spol. s.r.o.
- Legislativa.cz, s.r.o.
- Lékarna.cz (Pears Health Cyber, s.r.o.)
- Livesport s.r.o.
- Logio s.r.o.
- Mafra, a.s.
- ManpowerGroup s.r.o.
- Memsource a.s.
- METRANS, a.s
- MFL.cz s.r.o.
- MICROSOFT s. r. o.
- Minion Interactive s.r.o.
- Nano Energies Trade s. r. o.
- Navigo Solutions s.r.o.
- NESS Czech s.r.o.
- NetRex s.r.o.
- Nula s.r.o.
- OKsystem a.s.
- Onlio, a.s.
- Pixelmate s.r.o.
- POČÍTAČOVÁ POHOTOVOST s.r.o.
- Prague Labs s.r.o.
- PREGIS, a.s.
- ProductBoard s.r.o.
- PROFI-MEN, s.r.o.
- Prusa Research s.r.o.
- PŘEDVÝBĚR.CZ s.r.o.
- Q - COM, spol. s r.o.
- Quadient s.r.o
- R4U s.r.o.
- RCE systems s.r.o.
- REMANTE GROUP s.r.o.
- Reservio CZ, s.r.o.
- Revolt BI s.r.o.
- Rohlík.cz (Velká pecka s.r.o.)
- Sazka a.s.
- Scuk.cz (Lokalmarket s.r.o.)
- SEACOMP s.r.o.
- Seznam, a.s.
- SHERWOOD Digital a.s.
- Shoptet s.r.o.
- SII s.r.o.
- SiteOne, s.r.o.
- Slevomat s.r.o.
- Smartlook.com s.r.o.
- Solitea CDL, a.s.
- Solitea Česká republika, a.s.
- Sportisimo s.r.o.

- Stapro s.r.o.
- Talentica s.r.o.
- Tatum s.r.o.
- TechFides Solutions s.r.o.
- TESCO SW a.s.
- Tipsport a.s.
- TRAFICON TOBACCO RETAIL s.r.o.
- uLékaře.cz, s.r.o.
- Uloz.to (Uloz.to cloud s.r.o.)
- USU Software, s.r.o.
- Vodafone Czech Republic a.s.
- Web4U s.r.o.
- Webnode CZ s.r.o.
- Zásilkovna s.r.o.
- ZEEBRA Resource Solutions, s.r.o.

Příloha 2: Seznam dalších zmíněných technologií v inzerátech

- **Monitorovací a logovací nástroje:**
  - Grafana - grafana.com
  - Graylog - graylog.org
  - Jmetter - jmeter.apache.org
  - Logstash - elastic.co/logstash
- **Cachovací nástroje:**
  - Memcache - memcached.org
  - Redis - redis.io
- **Testovací a lintovací nástroje:**
  - Eslint - eslint.org
  - Mocha - mochajs.org
  - PHPstan - phpstan.org/user-guide/getting-started
  - PHPUnit - phpunit.de
  - Stylelint - stylelint.io
- **Nástroje a frameworky pro tvorbu UI:**
  - DOJO - dojotoolkit.org
  - Flutter - flutter.dev
  - Kendo UI - telerik.com/kendo-ui
- **Ostatní:**
  - Flux - facebook.github.io/flux
  - AWS - aws.amazon.com
  - Cassandra - cassandra.apache.org
  - Docker - docker.com
  - Elasticsearch - elastic.co
  - GraphQL - graphql.org
  - Kafka - kafka.apache.org
  - RabbitMQ - rabbitmq.com
  - Redux - redux.js.org
  - Selenium - selenium.dev
  - Traefik - doc.traefik.io
  - Vagrant - vagrantup.com

Příloha 3: Další vyučované technologie na školách

- Python - python.org
- WebGL - khronos.org/webgl
- Phaser - phaser.io
- Blazor - dotnet.microsoft.com/apps/aspnet/web-apps/blazor
- RPM - rpm.org
- Google Cloud - cloud.google.com
- Firebird - firebirdsql.org
- Codeigniter - codeigniter.com
- Strapi - strapi.io
- WordPress - cs.wordpress.org

Dotazník - vyučované technologie pro vývoj webových stránek

5 MINUT

Tento formulář slouží pro získání statistiky do diplomové práce ohledně vyučovaných technologiích pro vývoj webových stránek na školách.

I "nevím" a "nevyučujeme" se počítá 😐.

Pokud dotazník vyplníte a na konci zadáte e-mail, pošlu vám odkaz na tuto práci 🤩. Můžete si v ní následně o zmíněných technologiích přečíst a inspirovat se pro další výuku. Zadaný e-mail bude použit pouze pro zaslání odkazu na diplomovou práci.

V dotazníku jsou otázky s více možnostmi týkající se vyučovaných technologií. Můžete zadat i možnost, která v nabídce není.

🏠 Název školy

Jen pro vytřídění duplicit. Nebude uvedeno v práci.

Text stručné odpovědi

🏫 Typ školy *

1. Základní škola
2. Gymnázium
3. Střední škola
4. Vysoká škola

Má škola nějaké zaměření?

Např. Střední průmyslová škola | Obchodní akademie

Text stručné odpovědi

🌐 Webová adresa školy

Pro dohledání dalších informací o škole. Nebude uvedeno v práci.

Text stručné odpovědi

🗺️ Město / Obec, kde se škola nachází

Pro možnost vyvození dalších závěrů ze statistik.

Text stručné odpovědi

➗ Počet žáků (stačí odhad) *

- < 100
- > 100
- > 200
- > 300
- > 400
- > 500
- Jiná...

💻 Jazyky pro tvorbu webu *

- [ ] HTML + CSS
- [ ] JavaScript
- [ ] PHP
- [ ] ASP.NET (Core)
- [ ] Nevím 😕
- [ ] Jiná...

🖌️ CSS frameworky *

CSS framework je nějaká sada nastylovaných selektorů, které se přidávají do HTML a díky tomu upravují vzhled stránky.

- [ ] Bootstrap (https://getbootstrap.com/)
- [ ] Tailwind (https://tailwindcss.com/)
- [ ] Nevyučujeme
- [ ] Nevím 😕
- [ ] Jiná...

🔨 CSS preprocesory a postprocesory *

Preprocessory a postprocessory slouží k zjednodušení práce vývojáře. CSS se z nich teprve vytváří.

- [ ] SASS (https://sass-lang.com/)
- [ ] PostCSS (https://postcss.org/)
- [ ] Nevyučujeme
- [ ] Nevím 😕
- [ ] Jiná...

⚙️ JavaScriptové frameworky *

JavaScriptový framework je sada předpřipravené funkcionality pro vývojáře.

- [ ] jQuery (https://jquery.com/)
- [ ] Vue.js (https://vuejs.org/)
- [ ] AMP (https://amp.dev/)
- [ ] Nevyučujeme
- [ ] Nevím 😕
- [ ] Jiná...

Rozšíření a kompilátory jazyka JavaScript *

Rozšíření jazyka rozšiřují např. syntaxi jazyka. Kompilátory převádí kód využívající nové funkce do takové podoby, která funguje i ve starších prohlížečích.

- [ ] Typescript (https://www.typescriptlang.org/)
- [ ] Babel (https://babeljs.io/)
- [ ] Nevyučujeme
- [ ] Nevím 😕
- [ ] Jiná...

PHP frameworky *

PHP framework je sada předpřipravené funkcionality pro vývojáře.

- [ ] Symfony (https://symfony.com/)
- [ ] Nette (https://nette.org/cs/)
- [ ] Nevyučujeme
- [ ] Nevím 😕
- [ ] Jiná...

Balíčkovací nástroje *

Tyto nástroje spojují libovolné soubory v jeden.

- [ ] Rollup (https://rollupjs.org/guide/en/)
- [ ] Webpack (https://webpack.js.org/)
- [ ] Parcel (https://parceljs.org/)
- [ ] Nevyučujeme
- [ ] Nevím 😕
- [ ] Jiná...

HTTP servery *

- [ ] Nginx (https://www.nginx.com/)
- [ ] Node.js server (https://nodejs.org/en/docs/)
- [ ] PHP server
- [ ] ReactPHP (https://reactphp.org/)
- [ ] Apache (https://httpd.apache.org/)
- [ ] Nevyučujeme ☹
- [ ] Nevím 😕
- [ ] Jiná...

🕹️ Nástroje pro správu a instalaci balíčků *

Tyto nástroje slouží pro instalaci a správu balíčků pro zvolený programovací jazyk.

- [ ] NPM (https://www.npmjs.com/)
- [ ] YARN (https://yarnpkg.com/)
- [ ] Composer (https://getcomposer.org/)
- [ ] NuGet (https://www.nuget.org/)
- [ ] Nevyučujeme ☹️
- [ ] Nevím 😕
- [ ] Jiná...

💾 Prostory pro uložení projektu *

- [ ] GitHub (https://github.com/)
- [ ] Gitlab (https://gitlab.com/
- [ ] Azure DevOps (https://azure.microsoft.com/cs-cz/services/devops/)
- [ ] Nevyučujeme ☹️
- [ ] Nevím 😕
- [ ] Jiná...

🚀 Prostory pro spuštění webové aplikace *

- [ ] Netlify (https://www.netlify.com/)
- [ ] Vercel (https://vercel.com/)
- [ ] Webhostingy
- [ ] Nevyučujeme ☹️
- [ ] Nevím 😕
- [ ] Jiná...

🎓 Kde výše zmíněné technologie učíte? *

Můžete uvést obory a předměty. Oddělte je prosím čárkou.

- [ ] V běžné výuce (pokud jste škola se zaměřením na ICT, pak při výuce v daném oboru)
- [ ] Ve volitelných předmětech
- [ ] Jiná...

⏳ Kolik hodin výuky webových technologií studenti mají? *

Např. 2 h / týdně | 1 pololetí | 20 hodin celkem

Text stručné odpovědi

🤓 Co obsahuje výstupní zkouška *

- [ ] Písemný test
- [ ] Ústní zkouška
- [ ] Vytvoření projektu (se spuštěním na webhostingu)
- [ ] Vytvoření projektu (bez spuštění na webhostingu)
- [ ] Pouze za docházku
- [ ] Jiná...

😎 Vyučujete technologii, která v dotazníku není? Máte nějakou zajímavou poznámku? Pochlubte se.

Text dlouhé odpovědi

🤩 Chci si práci přečíst (zadejte e-mail)

Text stručné odpovědi