Česká zemědělská univerzita v Praze

Provozně ekonomická fakulta

Katedra informačních technologií

Bakalářská práce

Vývoj webových aplikací pomocí JavaScript frameworků

Lukáš Zikmund

ČESKÁ ZEMĚDĚLSKÁ UNIVERZITA V PRAZE

Provozně ekonomická fakulta

ZADÁNÍ BAKALÁŘSKÉ PRÁCE

Lukáš Zikmund

Informatika

Název práce

Vývoj webových aplikací pomocí JavaScript frameworků

Název anglicky

JavaScript frameworks for web application development

Cíle práce

Bakalářská práce je tematicky zaměřena na problematiku vývoje webových aplikací pomocí JavaScript (JS) frameworků.
Hlavním cílem práce je analyzovat a zhodnotit vybrané JS frameworky vhodné pro vývoj webových aplikací.
Dílčí cíle práce jsou:
- charakterizovat problematiku vývoje webových aplikací,
- zhodnocení vybraných JS frameworků,
- analyzovat možnosti využití JS frameworků pro tvorbu webových aplikací.

Metodika

Teoretická část bakalářské práce se bude zakládat na analýze a rešerši odborných zdrojů zaměřených na problematiku vývoje webových aplikací a JS frameworků.

V praktické části práce budou na základě poznatků zjištěných v analytické části zhodnoceny vybrané JS frameworky a formulovány možnosti využití jednotlivých frameworků pro vybrané typy webových aplikací.

Na základě syntézy teoretických a praktických poznatků budou zpracovány závěry bakalářské práce.

Doporučený rozsah práce

30-40 stran

Klíčová slova

React.js, JavaScript, HTML, CSS, framework, webová aplikace, skriptovací jazyk

---

Doporučené zdroje informací

GASSTON, P. Moderní web. Brno: Computer Press, 2015. ISBN 978-80-251-4345-2.

PEHLIVANIAN, A. – NGUYEN, D. JavaScript okamžitě. Brno: Computer Press, 2014. ISBN 978-80-251-4163-2.

THAU, D. Velký průvodce JavaScriptem : tvorba interaktivních webových stránek v praxi. Praha: Grada, 2009. ISBN 978-80-247-2211-5.

ŽÁRA, O. JavaScript : programátorské techniky a webové technologie. Brno: Computer Press, 2015. ISBN 978-80-251-4573-9.

Předběžný termín obhajoby

2023/24 LS – PEF

Vedoucí práce

Ing. Michal Stočes, Ph.D.

Garantující pracoviště

Katedra informačních technologií

Elektronicky schváleno dne 4. 7. 2023

doc. Ing. Jiří Vaněk, Ph.D.

Vedoucí katedry

Elektronicky schváleno dne 3. 11. 2023

doc. Ing. Tomáš Šubrt, Ph.D.

Děkan

V Praze dne 11. 03. 2024

Čestné prohlášení

Prohlašuji, že svou bakalářskou práci "Vývoj webových aplikací pomocí JavaScript frameworků" jsem vypracoval samostatně pod vedením vedoucího bakalářské práce a s použitím odborné literatury a dalších informačních zdrojů, které jsou citovány v práci a uvedeny v seznamu použitých zdrojů na konci práce. Jako autor uvedené bakalářské práce dále prohlašuji, že jsem v souvislosti s jejím vytvořením neporušil autorská práva třetích osob.

V Praze dne 13. 3. 2024

Poděkování

Rád bych touto cestou poděkoval Ing. Michalovi Stočesovi, Ph.D. za cenné rady, věcné připomínky a vstřícnost při konzultacích a vypracování bakalářské práce.

Vývoj webových aplikací pomocí JavaScript frameworků

Abstrakt

Bakalářská práce zkoumá problematiku vývoje webových aplikací, které jsou zaměřeny na interaktivitu s uživateli a využívají JavaScriptové frameworky. Hlavním cílem práce je hodnotit tyto frameworky na základě stanovených kritérií.

První část práce se zabývala charakteristikou vývoje webových aplikací. Tato část obsahovala popis klíčových technologií, které se využívají v průběhu vývoje webových aplikací, a detailní zkoumání procesů spojených s tímto odvětvím.

Druhá část byla zaměřena na samostatnou práci, kde bylo nejprve nutné vybrat vhodné JavaScriptové frameworky pro hodnocení. Výběr frameworků byl proveden srovnáním na základě dostupných dat. Následně byla vybrána kritéria na základě aspektů flexibility, udržitelnosti a bezpečnosti. Tyto kritéria byla použita k hodnocení vybraných frameworků metodou komparace.

Získané poznatky z vlastní práce sloužily k rozboru rozdílů mezi vybranými frameworky a umožňují dále zhodnotit jejich potenciál pro praktické využití.

Klíčová slova: React.js, Angular, Vue.js, HTML, CSS, JavaScript, framework, webová aplikace, skriptovací jazyk

JavaScript frameworks for web application development

Abstract

This bachelor thesis examines the development of web applications that prioritize interactivity with users and utilize JavaScript frameworks. The main objective is to evaluate these frameworks based on predetermined criteria.

The first section of the thesis covers the characteristics of web application development, including a description of key technologies and a detailed examination of associated processes.

The second part of the thesis focused on independent work. The first step was to select appropriate JavaScript frameworks for evaluation. The selection process involved comparing the available data and choosing frameworks based on their suitability for the project's needs. Next, criteria were selected based on the aspect of flexibility, sustainability, and security. These criteria were then used to evaluate the selected frameworks using a comparison method.

The actual work provided insight that were used to analyze the differences between the selected frameworks and evaluate their potential for practical use.

Keywords: React.js, Angular, Vue.js, HTML, CSS, JavaScript, framework, web application, scripting language

Obsah

1 Úvod

Díky sociálním sítím jako je Facebook, Instagram a Twitter spojuje internet lidi z celého světa. Nabízí široké spektrum služeb, které výrazně usnadňují každodenní život, jako jsou e-shopy Alza a Heureka nebo e-mailoví klienti například Seznam. Na internetu jsou velmi populární také zábavní a streamovací služby jako YouTube, Twitch a Netflix. Většina informací je dnes snadno dostupná pomocí různých vyhledávačů, které často odkazují na Wikipedii. Před použitím těchto služeb je nezbytné je nejprve vytvořit.

Vývoj webových aplikací souvisí s problematikou poskytování uživatelského zážitku. Existuje mnoho způsobů, jak tyto aplikace vyvíjet, ale v nejlepším případě by měly uživatelům dodat pocit spokojenosti a nechat je pohodlně používat aplikaci. V poslední době je kladen důraz právě na uživatelskou přívětivost a zážitek, zejména u jednostránkových aplikací, které jsou interaktivní a poskytují uživatelům dostatek informací o chodu aplikace a aktuálním dění.

V současné době je vývoj webových aplikací často založen na jazyce JavaScript, který umožňuje dynamickou úpravu obsahu stránky bez nutnosti opakovaného načítaní. Přestože je JavaScript vhodný pro interaktivitu, vývoj složitých aplikací může být obtížný a nákladný. Proto se často používají různé nástroje a pomocné prostředky, které vývoj usnadňují. Tyto nástroje a knihovny umožňují vytvoření frameworku, který poskytuje vývojářům prostředí pro implementaci konkrétních potřeb aplikace, aniž by se museli zabývat základními aspekty vývoje. S narůstajícím množstvím těchto frameworků je výběr vhodné varianty stále složitější, což vyžaduje důkladné posouzení potřeb aplikace a vývojáře.

Cílem bakalářské práce je podrobněji představit problematiku vývoje webových aplikací napsaných pomocí JavaScriptových frameworků. Hlavní zaměření práce spočívalo v rozhodování mezi různými variantami těchto frameworků s využitím doporučené literatury, odborných zdrojů a praktických zkušeností odborníků z praxe.

2 Cíl práce a metodika

2.1 Cíle práce

Bakalářská práce je tematicky zaměřena na problematiku vývoje webových aplikací pomocí JavaScript (JS) frameworků.

Hlavním cílem práce je analyzovat a zhodnotit vybrané JS frameworky vhodné pro vývoj webových aplikací.

Dílčí cíle práce jsou:

- charakterizovat problematiku vývoje webových aplikací,
- zhodnocení vybraných JS frameworků,
- analyzovat možnosti využití JS frameworků pro tvorbu webových aplikací.

2.2 Metodika

Teoretická část bakalářské práce se bude zakládat na analýze a rešerši odborných zdrojů zaměřených na problematiku vývoje webových aplikací a JS frameworků.

V praktické části práce budou na základě poznatků zjištěných v analytické části zhodnoceny vybrané JS frameworky a formulovány možnosti využití jednotlivých frameworků pro vybrané typy webových aplikací.

Na základě syntézy teoretických a praktických poznatků budou zpracovány závěry bakalářské práce.

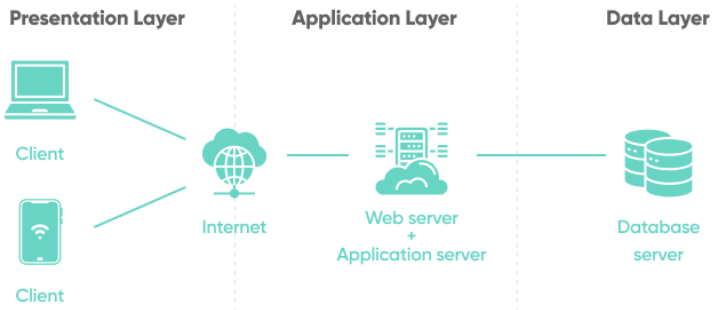
3 Teoretická část

Tato kapitola zkoumá teoretická východiska a provádí rešerši odborných pramenů. Tyto informace přispívají k hlubšímu pochopení problematiky vývoje webových aplikací, zejména s ohledem na technologie spojené se skriptovacím jazykem JavaScript.

3.1 Webová aplikace

Webová aplikace je specifický typ aplikačního programu, který má několik význačných rysů a výhod. Za prvé, webová aplikace nepotřebuje být nainstalována na konkrétním zařízení, jako jsou počítače, notebooky, chytré telefony nebo tablety. To znamená, že uživatelé nemusí provádět složité instalační procesy nebo zabírat místo na svých zařízeních. Místo toho mohou přistupovat k webové aplikaci prostřednictvím svých internetových prohlížečů. (1)

Webová aplikace je uložena na vzdáleném serveru, což je počítač, který je připojen k internetu a poskytuje služby a data. Tento server obsahuje veškerý potřebný kód a data pro fungování aplikace. Uživatelé přistupují k webové aplikaci jednoduše tím, že zadají její internetovou adresu do svého prohlížeče. Prostřednictvím této adresy se připojí ke vzdálenému serveru, kde je aplikace hostována (viz Obrázek 1). (1) (2)



Obrázek 1- Architektura webové aplikace

zdroj: https://www.softkraft.co/web-application-architecture/

Webové stránky slouží jako rozhraní mezi uživatelem a webovou aplikací. Uživatelé interagují s aplikací pomocí tlačítek, formulářů, odkazů a dalších prvků na webových stránkách. Výhodou tohoto modelu je, že uživatelé mohou přistupovat k webovým aplikacím z libovolného zařízení s internetovým připojením a webovým prohlížečem, což zajišťuje vysokou flexibilitu a dostupnost. (1) (2)

Celkově lze říci, že webové aplikace přinášejí uživatelům pohodlí a snadný přístup k aplikacím a službám bez nutnosti instalace a údržby na jejich zařízení. Tyto aplikace jsou často využívány pro různé účely, jako jsou sociální sítě, e-commerce platformy, online hry a mnoho dalších. (2)

3.1.1 Technologie používané při vývoji webových aplikací

Vytváření složitých webových aplikací zahrnuje využití technologií jak se strany serveru, tak ze strany klienta. Když se mluví o technologiích webového vývoje, obvykle jsou tím myšleny technologie na straně klienta, které se používají k sestavení a zobrazení všeho, s čím konečný uživatel pracuje. Hlavním jazykem pro vývoj webu je JavaScript, přítomný na většině webových stránek a aplikací na straně klienta. To znamená, že webové technologie jsou převážně založené na JavaScriptu, kdežto na straně serveru jsou technologie různé. Mezi nejpoužívanější patří .NET, Java, PHP, Ruby, Python nebo framework Node.js. (3)

3.1.2 Priority při tvorbě webových stránek

Trendy, které vývojáři aplikací upřednostňují během posledních několika let:

Vývoj pro mobilní telefony – není tajemstvím, že lidé k prohlížení internetu používají více telefony než počítače, proto vývojáři musejí vytvářet responzivní aplikace, které se přizpůsobí velikosti obrazovky. Dále se soustředí na optimalizaci aplikací pro mobilní zařízení. (4)

Bezpečnost – čím více lidí přistupuje k webu skrz mobilní zařízení, tím má bezpečnost nejvyšší prioritu. Vývojáři musí chránit aplikace před potenciálním nebezpečím. To znamená, že vývojáři musejí být informovaní o webové bezpečnosti. (4)

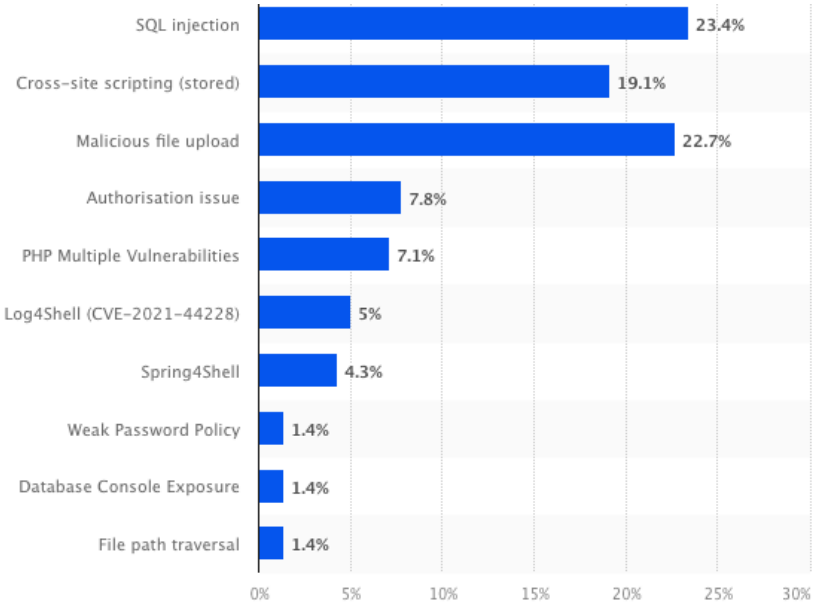
Přístupnost – vývojáři by měli zajistit přístup pro všechny uživatele bez ohledu na to, zda mají nějaké postižení, používají jiný prohlížeč nebo pochází z odlišných kultur. (4)

Umělá inteligence – umělá inteligence dnes hraje ve vývoji webových aplikací důležitou roli. Vývojáři ji využívají k optimalizaci webového obsahu a tím zlepšují zážitek pro uživatele. (4)

3.1.3 Zabezpečení webových stránek

Testování webové bezpečnosti cílí na hledání bezpečnostních slabin ve webových aplikacích a jejich konfiguraci. Hlavní cílem je aplikační vrstva (např. na čem je spuštěn HTTP protokol). Testování bezpečnosti zahrnuje mimo jiné odesílání různých typů vstupů za účelem vyvolání chyb a přimění systému chovat se nepředvídatelně. Tyto takzvané “negativní testy“ zkoumají, zda systém nedělá něco, k čemu není určen. Je také důležité pochopit, že testování webové

bezpečnosti není jen o testování bezpečnostních funkcí (např. ověření a autorizace), které mohou být implementovány v aplikaci. Stejně důležité je otestovat, zda jsou ostatní funkce implementovány bezpečným způsobem (např. použití správné kontroly vstupů a kódování výstupů). (5) Nejčastěji používaným útokem je SQL Injection (viz Obrázek 2). Jedná se o techniku vkládání kódu, která může zničit databázi. Spočívá v umístění škodlivého kódu do příkazů SQL prostřednictvím vstupu na webové stránce. (6)



Obrázek 2 - Rozložení útoků na webové aplikace v roce 2023

zdroj: https://www.statista.com/statistics/806081/worldwide-application-vulnerability-taxonomy/

3.2 Technologie použité při tvorbě ukázkové aplikace

Při vytváření webových aplikací se využívají následující technologie: HTML, CSS, JavaScript. Tyto technologie jsou základem pro vývoj moderních webových aplikací a internetových prezentací. Jsou široce používané a poskytují základní stavební kameny pro vytvoření uživatelsky přívětivého prostředí na webu. Jedním z významných faktorů je, že všechny tyto technologie jsou volně dostupné a zdarma, což usnadňuje jejich širokou adopci.

Během vývoje aplikace jsou využívány nejnovější funkce a možnosti, které HTML5 a CSS3 nabízejí. Tyto novinky poskytují vývojářům širší spektrum nástrojů pro vytvoření interaktivních a esteticky přitažlivých webových rozhraní. Díky tomu jsou k dispozici moderní technické prostředky, které vývojářům usnadňují práci při tvorbě webový aplikací a zároveň zlepšují uživatelský zážitek.

3.2.1 HTML

HTML (HyperText Markup Language) je značkovací jazyk, který definuje strukturu obsahu. HTML se skládá z řady prvků, které se používají k obklopení nebo obalení různých částí obsahu tak, aby vypadal určitým způsobem nebo se určitým způsobem choval. Obklopující značky mohou způsobit, že slovo nebo obrázek bude hyperaktivně odkazovat na jiné místo, slova budou napsaná kurzívou, bude možné zvětšit nebo zmenšit písmo atd. (7)

HTML prvek je definovaný začáteční značkou neboli tagem, určitým obsahem a ukončující značkou. Tyto prvky se nazývají párové a mohou vypadat takto: `<p>Nějaký odstavec.</p>`. Tagy jsou označeny špičatými závorkami (<>), mezi které se vkládá název a případné vlastnosti. Některé HTML značky nenesou žádný obsah. Těmto značkám se říká nepárové značky. Nepárové značky nemají ukončující tag a jedná se například o: `<img>`. (8)

HTML5 vzniklo jako nástupce specifikace HTML 4.01 a přineslo mnoho nových funkcí, upravilo chování některých existujících prvků a další byly označeny za zastaralé. Standardizuje návrhové vzory naučené vývojáři a rozšiřuje jazyk HTML tak, aby lépe vyhovoval moderním požadavkům. Nové funkce HTML5 umožňují efektivnější strukturování dokumentů s ohledem na sémantiku a přístupnost a zahrnují i širokou škálu nových formulářových a aplikačních prvků. Dále HTML5 přináší nativní podporu pro přehrávání videí a bylo vytvořeno dvěma hlavními skupinami vývojářů: WHATWG (Web Hypertext Application Technology Working Group), která vytváří dynamickou specifikaci HTML bez číselných verzí, a konsorciem W3C (World Wide Web Consortium), které vytváří číslované verze specifikace pro zajištění kompatibility mezi prohlížeči. (9)

Šablona dokumentu HTML5 se mírně změnila od předešlých specifikací. Přestože se nejedná o velké změny, tak stojí za zmínění. Uvedený blok kódu představuje základní šablonu HTML5 (viz Obrázek 3). (9)

```
<!DOCTYPE html>
<html lang="cs">
<head>
  <meta charset="utf-8">
  <title></title>
</head>
<body></body>
</html>
```

Obrázek 3 - Základní šablona HTML5

První změna se týká deklarace typu dokumentu na prvním řádku (DOCTYPE). Jedná se o pozůstatek, který se musel v minulosti psát – HTML Strict, HTML Transitional, XHTML 1.1 atd. V HTML5 to není nutné, protože existuje jen jedna varianta, takže deklarace typu dokumentu nemusí být tak detailní. (9)

Druhým bodem výše zmíněného kódu je element meta, pomocí něhož definujeme znakovou sadu UTF-8, což je nejrozšířenější znaková sada na webu. U prvku meta je přidán atribut charset: `<meta charset=“utf-8“>`. (9)

Kromě změn v šabloně je důležité zmínit dva osvědčené postupy v praxi. HTML5 si klade za cíl zjednodušit psaní a údržbu kódu tím, že umožňuje vynechat atribut type při načítání externích prostředků. Dříve bylo nutné v HTML 4.01 a XHTML definovat typ pro všechny prvky link, script a style, ale nyní lze tento atribut vynechat, což přispívá k čistšímu kódu. HTML5 má také velmi tolerantní syntaxi, která ignoruje rozdíly mezi velkými a malými písmeny, zjednodušuje zápis hodnot atributů a umožňuje vynechání uzavíracích značek u prázdných elementů. Některé atributy mají pouze hodnoty true nebo false, takže pouze jejich přítomnost zastává pravdivé hodnoty. (9)

Pro vytvoření kvalitní webové stránky je klíčová správná struktura dokumentu HTML, která poskytuje základní rámec pro veškerý obsah. Bez ohledu na typ webu, ať už se jedná o interaktivní aplikaci, hybridní mobilní rozhraní nebo jednoduché webové stránky, je solidní struktura nezbytná pro přístupnost a udržitelnost. Dobře navržené stránky jsou lépe interpretovány prohlížeči a umožňují efektivnější zpracování a menší paměťovou náročnost. Jednoduchá, ale významná struktura umožňuje rychlejší vyhledávání a lepší pochopení obsahu vyhledávačům a dalším agentům. S využitím HTML5 a přidružených technologií lze snadno vytvořit spolehlivé, srozumitelné a efektivní stránky s bohatým obsahem. (9)

Jednou z hlavních novinek v jazyce HTML5 jsou nové sémantické prvky, které rozšiřují možnosti označování obsahu. Tyto prvky umožňují vývojářům lépe strukturovat a označovat různé části webu, což snižuje potřebu používání obecných prvků div s atributy id nebo class. Specifikace od konsorcia W3C obsahuje celkem deset strukturálních prvků, z nichž sedm je nových a tři byly zachovány ze starších verzí. Mezi novými prvky jsou i rozdělující elementy, které pomáhají oddělit různé části obsahu. (9)

Patří mezi ně:

- Article – nezávislá část dokumentu. Například příspěvek diskusního fóra, příspěvek blogu nebo uživatelský komentář.
- Aside – oblast stránky, která okrajově souvisí s okolním obsahem, ale je možné ji považovat za nezávislou. Například postranní sloupek v novinovém článku.
- Nav – navigační oblast dokumentu, která obsahuje odkazy na další dokumenty nebo oblasti stejného dokumentu.
- Section – tematické seskupení obsahu. Například kapitola v knize, karta v dialogovém okně nebo úvod na domovské stránce. (9)

Zbylé tři prvky definují oblasti uvnitř rozdělení obsahu:

- Footer – zápatí dokumentu nebo jeho oblasti. Nejčastěji obsahuje metadata o sekci, ve které se nachází; např. údaje o autorovi.
- Header – záhlaví dokumentu nebo jeho oblasti. Většinou obsahuje nadpisy.
- Hgroup – seskupuje více nadpisů, např. nadpis s podnadpisem nebo seznamem štítků. (9)

Účelem těchto nových prvků je usnadnit strukturování dokumentů tak, aby je webové prohlížeče a další programy mohly lépe interpretovat. Tato struktura pomáhá například předčítačům obrazovek při čtení obsahu pro zrakově postiženého uživatele. Nové prvky vytvářejí mapy dokumentů, ukazují hierarchii obsahu, důležitost různých částí, vztahy mezi nimi a další. V HTML4 byly tyto informace závislé na nadpisech (h1-h6), kde h1 představoval nadpis stránky a následné nadpisy označovaly hierarchii. V hTML5 je struktura dokumentu explicitně definována pomocí rozdělujících prvků, nikoliv pouze nadpisů. Tento koncept se nazývá explicitní rozčleňování. (9)

„*Proč vůbec ztrácet čas se sémantikou?*“ Existují dva dobré důvody, proč používat sémantické prvky. První je ten, že napsaný kód splňuje standard a vylepšuje to jeho udržitelnost. Při použití sémantických prvků v kódu bude daleko snazší a rychlejší se orientovat. Druhý důvod je méně jasný a spočívá v tom, že sémantické prvky zvyšují *aboutness* (český doslovný překlad je o čem-ství). (9)

Správné označení obsahu na webových stránkách je klíčové pro poskytnutí sémantického kontextu. Sémantické prvky HTML5 umožňují strukturovat obsah tak, aby byl srozumitelný nejen lidem, ale i automatizovaným systémům jako jsou roboti. To může být dosaženo pomocí

mikroformátů, RDFa (Resource Description Framework in attributes) nebo mikrodat. HTML5 zavedlo jednoduchou syntaxi mikrodat, která umožňuje definovat strojově čitelná data pomocí dvojic název-hodnota. Atribut itemscope určuje oblast působení dané položky, zatímco hodnota atributu itemprop představuje název vlastnosti a obsah prvku je její hodnotou. (9)

3.2.2 CSS

Kaskádové styly, zkráceně CSS, představují jazyk určený k definici vzhledu dokumentů napsaných v HTML nebo XML, včetně souvisejících dialektů XML jako jsou SVG, MathML a XHTML. Jejich účelem je specifikovat, jak mají být prvky zobrazeny na různých médiích včetně obrazovek a tisku. CSS je základním jazykem pro tvorbu webových stánek a je standardizován podle směrnic W3C ve všech webových prohlížečích. Dříve byl vývoj jednotlivých částí specifikace synchronní, což umožňovalo označování verzí nejnovějších doporučení. Nicméně v současnosti jsou specifikace včetně CSS3, označovány jednoduše jako CSS bez přiloženého čísla verze. (10)

Po vydání CSS 2.1 se rozsah specifikace výrazně zvětšil a pokrok v jednotlivých modulech CSS se začal natolik lišit, že bylo efektivnější vyvíjet a vydávat doporučení pro každý modul zvlášť. Namísto verzování specifikace CSS nyní W3C pravidelně vydává nástin posledního stabilního stavu specifikace CSS a pokroku jednotlivých modulů. Moduly CSS nyní mají čísla verzí nebo úrovně, například CSS Color Module Level 5. (10)

CSS3, zkráceně třetí verze kaskádových stylů, představuje evoluci standardu CSS určeného pro formátování a stylování webových stránek. Obsahuje základní standardy CSS2 a řadu úprav a vylepšení. Oproti předchozí verzi se CSS3 nevztahuje k jedné masivní specifikaci, nýbrž je rozděleno do jednotlivých modulů, což usnadňuje jeho implementaci a rozvoj. Tento přístup umožňuje postupné vydávání specifikací, přičemž některé moduly jsou stabilnější a jiné jsou stále ve vývoji. Některé z těchto modulů se již blíží k oficiálnímu doporučení, zatímco jiné jsou stále ve fázi rozpracovaných návrhů, z nichž některé byly představeny již v roce 1999 (11)

Výběr hlavních modulů v CSS3:

- Box model
- Hodnoty obsahu a nahrazujícího obsahu
- Textové efekty
- Selektory

- Pozadí a okraje
- Animace
- Uživatelské rozhraní (UI)
- Rozložení více sloupců
- 2D/3D transformace (11)

3.2.3 JavaScript

JavaScript je lehký programovací jazyk, který vývojáři webových stránek běžně používají k přidávání dynamických interakcí do webových stránek, aplikací, serverů a dokonce i her. Bez problémů funguje vedle HTML a CSS, doplňuje CSS při formátování prvků HTML a zároveň poskytuje interakci s uživatelem, což je schopnost, kterou samotné CSS neumí. Vzhledem k rozšířenému používání jazyka JavaScript při vývoji webových stránek, mobilních aplikací a her se jedná o cenný jazyk, který stojí za to si osvojit. (12)

Původní verze skriptovacího jazyka byly zpočátku určeny pouze pro interní použití. Poté, co společnost Netscape předložila jazyk organizaci ECMA International jako standardní specifikaci pro webové prohlížeče, se JavaScript stal průkopníkem vydání ECMAScriptu. Tento univerzální skriptovací jazyk měl za cíl zajistit interoperabilitu webových stránek napříč různými prohlížeči a zařízeními. JavaScript se od té doby neustále rozvíjí s nástupem nových prohlížečů, jako je Mozilla Firefox nebo Google Chrome. Chrome dokonce vyvinul první moderní JavaScriptový engine nazvaný V8, který kompiluje bajtový kód do nativního strojového kódu. Dnes má JavaScript k dispozici širokou škálu frameworků a knihoven, které usnadňují vývoj komplexních projektů, jako jsou Angular, jQuery a React. Po zavedení Node.js se JavaScript, původně určený pro běh na straně klienta, rozšířil i na stranu serveru – multiplatformní serverové prostředí postavené na enginu V8 JavaScriptu používaném v prohlížeči Google Chrome. Ačkoli se primárně zaměřuje na webové aplikace, programovací možnosti JavaScriptu jsou dnes využívány i v dalších oblastech. (12)

JavaScript se odlišuje od ostatních programovacích jazyků v několika ohledech. Za prvé, je interpretovaným jazykem, což znamená, že lze kód spustit přímo v prohlížeči. Na rozdíl od kompilovaných jazyků, jako jsou C, C++ a Java, není potřeba JavaScript před spuštěním překládat do strojového kódu. Dále využívá dynamické typování, které umožňuje propojení typů proměnných s hodnotami za běhu. To usnadňuje psaní kódu, protože není nutné deklarovat typy proměnných. JavaScript se především používá při spouštění na straně klienta, což

znamená, že je spuštěn ve webovém prohlížeči uživatele, a ne na serveru. Tím má možnost interagovat s uživatelem, reagovat na jeho vstupy a dynamicky aktualizovat obsah stránky bez nutnosti komunikace se serverem. Nakonec JavaScript je všudypřítomným jazykem, který není omezen na konkrétní případy použití a podporují ho všechny hlavní prohlížeče. (13)

JavaScript má řadu výhod, díky kterým je lepší volbou než jeho konkurenti. Zde je seznam několika výhod:

- Jednoduchost – díky jednoduché struktuře se JavaScript snadněji učí, implementuje a také běží rychleji než některé jiné jazyky.
- Rychlost – JavaScript spouští skripty přímo ve webovém prohlížeči, aniž by se nejprve připojoval k serveru nebo potřeboval překladač. Většina hlavních prohlížečů navíc umožňuje kompilovat kód JavaScriptu během spouštění programu.
- Univerzálnost – JavaScript je kompatibilní s dalšími jazyky, jako jsou PHP, Perl a Java. Vývojářům také zpřístupňuje datovou vědu a strojové učení.
- Oblíbenost – k dispozici je spousta zdrojů a fór, které pomáhají začátečníkům s omezenými technickými dovednostmi a znalostmi jazyka JavaScript.
- Zatížení serveru – další výhodou práce na straně klienta je, že JavaScript snižuje požadavky odesílané na server. Ověřování dat lze provádět prostřednictvím webového prohlížeče a aktualizace se týkají pouze určitých částí webové stránky.
- Aktualizace – vývojový tým JavaScriptu a ECMA International neustále aktualizuje a vytváří nové rámce a knihovny, čímž zajišťuje jeho aktuálnost v rámci odvětví. (12)

Stejně jako každý jiný programovací jazyk má i JavaScript několik omezení, která je třeba brát v úvahu. Níže jsou uvedeny některé z nevýhod používání jazyka JavaScript:

- Kompatibilita s prohlížeči – různé webové prohlížeče interpretují kód JavaScriptu odlišně, což způsobuje nekonzistenci. Proto byste měli svůj skript otestovat ve všech populárních webových prohlížečích, včetně jejich starších verzí, aby nedošlo k poškození uživatelského prostředí.
- Bezpečnost – kód JavaScriptu, který je spuštěn na straně klienta, je zranitelný vůči zneužití nezodpovědnými uživateli.
- Ladění – některé editory HTML sice podporují ladění, ale jsou méně efektivní než jiné editory. Vzhledem k tomu, že prohlížeče nezobrazují žádná upozornění na chyby, může být nalezení problému náročné. (12)

React.js, často nazývaný jednoduše React, je framework určený k tvorbě uživatelských rozhraní. Každá webová aplikace postavená na Reactu je složena z opakovaně využitelných komponent, které představují různé části uživatelského rozhraní – například navigační panel, patička nebo hlavní obsah. Tento přístup k vývoji zjednodušuje proces, protože eliminuje opakující se kód. Stačí vytvořit jednou logiku a poté příslušnou komponentu naimportovat do libovolné části kódu, kde je potřeba. React se používá k vývoji jednostránkových aplikací, což znamená, že místo tradičního načítání nové stránky při každém vykreslení, React načítá obsah přímo z komponent. Tento přístup vede k rychlejšímu vykreslení stránky bez nutnosti opakovaného načítání. (14)

Mnoho vývojářů a organizací se rozhodlo pro React místo jiných knihoven nebo frameworků z několika důvodů. Prvním z nich je jeho snadné naučení – React disponuje solidní dokumentací a nabízí mnoho bezplatných zdrojů, které vytvořili ostatní vývojáři a jsou dostupné online. Dalším důležitým faktorem jsou opakovaně použitelné komponenty. Každá komponenta v Reactu má vlastní logiku, kterou lze opakovaně použít kdekoliv v aplikaci, což snižuje nutnost opakovaného psaní stejného kódu. Dále je React vyhledávanou dovedností na trhu práce, jelikož mnoho pracovních pozic pro vývoj frontendových webových aplikací vyžaduje znalost Reactu. Dalším přínosem je vyšší výkon – díky virtuálnímu DOM (Document Object Model) Reactu je vykreslování stránek rychlejší, což umožňuje rychlou navigaci mezi stránkami bez nutnosti znovunačtení celé stránky. V neposlední řadě, široká rozšiřitelnost Reactu umožňuje vývojářům vybírat si z různých nástrojů a knihoven, které se specializují na různé části vývoje aplikace, jako jsou knihovny pro směrování nebo nástroje pro návrh uživatelského rozhraní. Přestože je React populární nástroj pro vytváření uživatelského rozhraní, stále jsou důvody, proč se někteří vývojáři nebo začátečníci rozhodnout jej nepoužívat. Velkou překážkou pro začátečníky je obtížné pochopení Reactu, pokud nemají dobré znalosti JavaScriptu (zejména specifikace ES6). Další nevýhoda Reactu je, že je dodáván bez některých běžných funkcí jako správa jednoho stavu a směřování. (14)

Angular je open-source JavaScriptový framework, vyvíjený a udržovaný společností Google, který poskytuje standardní strukturu a další funkce pro zjednodušení vývoje webových a mobilních aplikací. Jedním z charakteristických rysů Angularu je jeho použití běžného DOM spolu s TypeScriptem – staticky typovanou nadstavbou jazyka JavaScript – ke zlepšení čitelnosti a udržitelnost kódu. Tento framework přináší do aplikací dvousměrnou vazbu dat, což dělá bezproblémové spojení mezi modelem (daty) a pohledovými komponentami. Tato automatická synchronizace umožňuje snadnou manipulaci s prvky webové stránky, aniž by

bylo nutné pro ni psát další kód. Angular se v průběhu času vyvíjel s vydáním mnoha verzí. Je široce využíván společnostmi po celém světě díky svým modulárním službám, kompatibilitě napříč platformami, testovatelnosti a mnoha dalším výhodám. (15)

Vue.js je lehký JavaScriptový framework určený k vytváření reaktivních uživatelských rozhraní pro webové aplikace. Vue.js rozšiřuje standardní HTML a CSS o soubor výkonných nástrojů pro vytváření frontendu interaktivních webových aplikací. Působící v rámci frameworku model-pohled-viewmodel se hlavní knihovna Vue.js zaměřuje na vrstvu viewmodel, poskytující reaktivní synchronizaci mezi modelovou a zobrazenou vrstvou prostřednictvím dvousměrného vázaní dat. To zjednodušuje tvorbu moderních jednostránkových aplikací. Vazba dat umožňuje Vue.js dynamicky aktualizovat HTML prvky, které jsou „vázány“ na podkladové objekty Vue.js. To umožňuje vývojářům vytvářet webové aplikace, které mohou běžet v prohlížeči uživatele a poskytovat interaktivní zážitek, který nevyžaduje obnovení stránky. Se schopností aktualizovat HTML a CSS zobrazené v prohlížeči v reakci na události v podkladovém JavaScriptu, mohou vývojáři pomocí Vue.js vytvářet od živých chatů až po animované videohry v rámci webového prohlížeče. (16)

3.2.4 Optimalizace pro vyhledávače

SEO (Search Engine Optimalization) je soubor procesů zaměřených na zlepšení viditelnosti webové stránky ve vyhledávačích jako je Google, s cílem získat více organického provozu. SEO se zabývá uspokojováním potřeb uživatelů vyhledávání tím, že vytváří relevantní, vysoce kvalitní obsah a poskytuje co nejlepší uživatelskou zkušenost. Aktivity SEO mohou probíhat jak na stránce (on-site), tak mimo ni (off-site). Proto lze často vidět SEO rozděleno na „on-page“ a „off-page“. V praxi zahrnuje SEO obvykle: zkoumání klíčových slov, tvorbu a optimalizaci obsahu, technickou optimalizaci a budování zpětných odkazů. (17)

3.2.5 Validace dat

Validace dat je proces kontroly a ověřování, zda data zadaná uživateli nebo přijatá z jiných zdrojů splňují očekávaná kritéria a formát. Validace dat může být prováděna na různých úrovních, jako je validace na straně klienta, na straně serveru nebo na úrovni databáze. Validace na straně klienta zahrnuje použití JavaScriptu nebo atributů HTML5 k ověření dat před jejich odesláním na server. To může zlepšit zpětnou vazbu uživatelů a snížit provoz sítě, ale nemůže zaručit integritu nebo bezpečnost dat, protože je možné ji obejít nebo manipulovat zlomyslnými uživateli. Validace na straně serveru zahrnuje použití programovacího jazyka, jako je PHP,

Python nebo Ruby k ověření dat po jejich přijetí od klienta. To může zajistit konzistenci a bezpečnost dat, protože může zabránit SQL injekci nebo jiným útokům, které by mohly zneužít databázi. Validace na úrovni databáze zahrnuje použití omezení, spouštěcích mechanismů nebo uložených procedur k ověření dat před jejich vložením, aktualizací nebo smazáním z databáze. To zajišťuje integritu a přesnost dat, protože může zabránit anomáliím dat nebo porušení schématu databáze. (18)

3.2.6 Debuging

Debugování je proces počítačového programování pro nalezení a odstranění chyb v softwaru nebo na webových stránkách. Často vyžaduje komplexní postup k identifikaci důvodu, proč chyba nastala a vyvinutí strategií, které zajistí, že program může v budoucnosti běžet hladce pro uživatele. Softwaroví vývojáři a inženýři často ladí programy pomocí digitálního nástroje k prohlížení a úpravě kódovacího jazyka, který obsahuje instrukce, jak program funguje. Debugování jim umožňuje adresovat jednotlivé části kódu, aby se zajistilo, že každá část programu funguje optimálním způsobem tak, jak se očekává. (19)

4 Praktická část

Praktická část vychází z teoretické části a zaměřuje se na výběr a následnou analýzu JavaScriptových frameworků. Postupně bylo provedeno zhodnocení tří vybraných frameworků podle stanovených kritérií, u kterých byly porovnány následující funkcionality: uživatelská zkušenost, optimalizace pro prohlížeče, validace vstupů a debuging.

4.1 Výběr frameworků ke zhodnocení

Kapitola se zaměřuje na proces výběru konkrétních JavaScriptových frameworků, přičemž klade důraz na několik klíčových kritérií:

Prvním z těchto kritérií byla *historie na trhu*. Bylo nezbytné, aby vybrané frameworky nebyly jen aktuální, ale také aby měly dlouhou a aktivní přítomnost na trhu. To poskytlo ujištění o jejich dlouhodobé životaschopnosti a schopnosti přizpůsobit se dynamickému prostředí vývoje webových aplikací. Dlouhodobá existence na trhu byla vnímána jako důležitý indikátor stability a důvěryhodnosti daného frameworku.

Druhým kritériem bylo sledování *aktivního používání frameworku v praxi a jejich osvědčení v reálných projektech*. Toto kritérium poskytlo důležitý pohled na to, jak jsou frameworky integrovány do existujících projekčních scénářů a jak si vedou ve skutečném prostředí vývoje. Osvědčení praxí bylo nezbytné pro zajištění, že vybrané nástroje nejsou pouze teoreticky robustní, ale také prakticky účinné a odpovídají potřebám reálného světa.

Dalším kritériem při výběru byla *udržitelnost a aktivita komunity*. Vybrané frameworky by měly prokázat pravidelný a aktivní vývoj ze strany komunity. Tato aktivita naznačuje schopnost frameworků pružně reagovat na nové vývojové trendy a předvídat potřeby vývoje webových aplikací v budoucnu. Udržitelnost je zde chápána nejen jako pravidelné aktualizace a opravy, ale také jako schopnost inovace a přizpůsobení se měnícím se požadavkům odvětví.

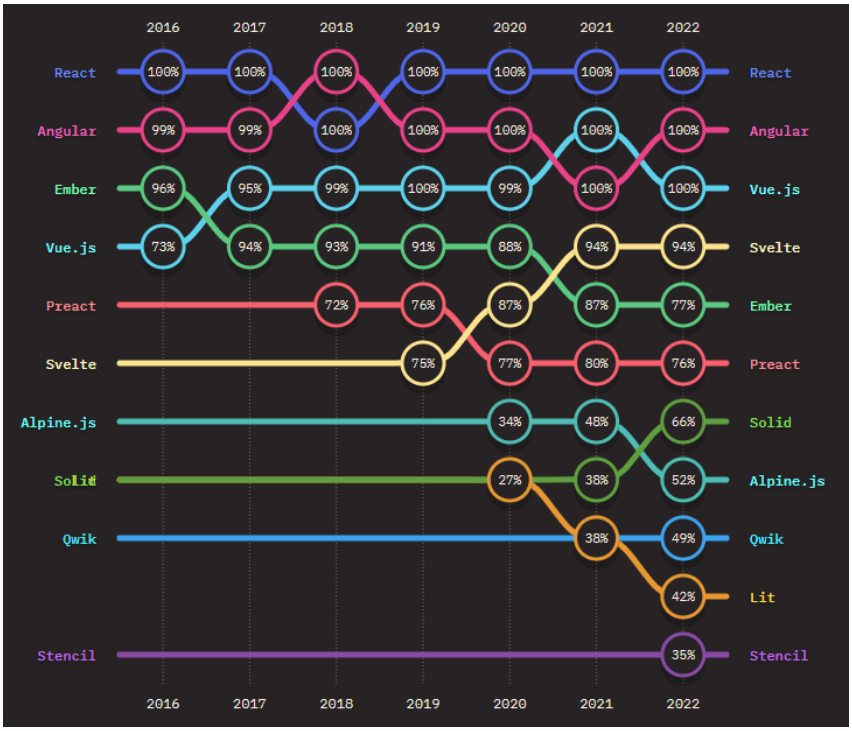
Posledním kritériem byla *open-source povaha* frameworků, přičemž byla dána přednost těm s MIT licencí. Open-source umožňuje transparentnost, flexibilitu a spolupráci komunity, což jsou důležité prvky při vývoji frameworků. Licence MIT pak poskytuje benevolentní podmínky pro svobodné využití, distribuci a modifikaci kódu, což je v souladu s filozofií otevřenosti a podporuje svobodu využívání frameworků. Tato kombinace kritérií zajišťuje, že vybrané frameworky nejen odpovídají na technologické nároky a osvědčení praxí, ale také jsou

v souladu s principy udržitelnosti, aktivním přístupem komunity a otevřeným přístupem k vývoji softwaru.

4.1.1 Kritéria výběru frameworku

Praktická část byla směřována na posouzení tří frameworků. V souladu s předem stanovenými podmínkami bylo nezbytné stanovit kritéria, která omezila výběr na tři konkrétní frameworky. Tento výběr kritérií a frameworků byl proveden na základě průzkumu State of JS 2022 (v době psaní této práce nebyly zpracovány výsledky z roku 2023) a analýzy dat z webového nástroje npm trends. Tyto informace byly důležité pro získání přesného obrazu o aktuálním stavu a preferencích ve vývoji JavaScriptových frameworků v daném období.

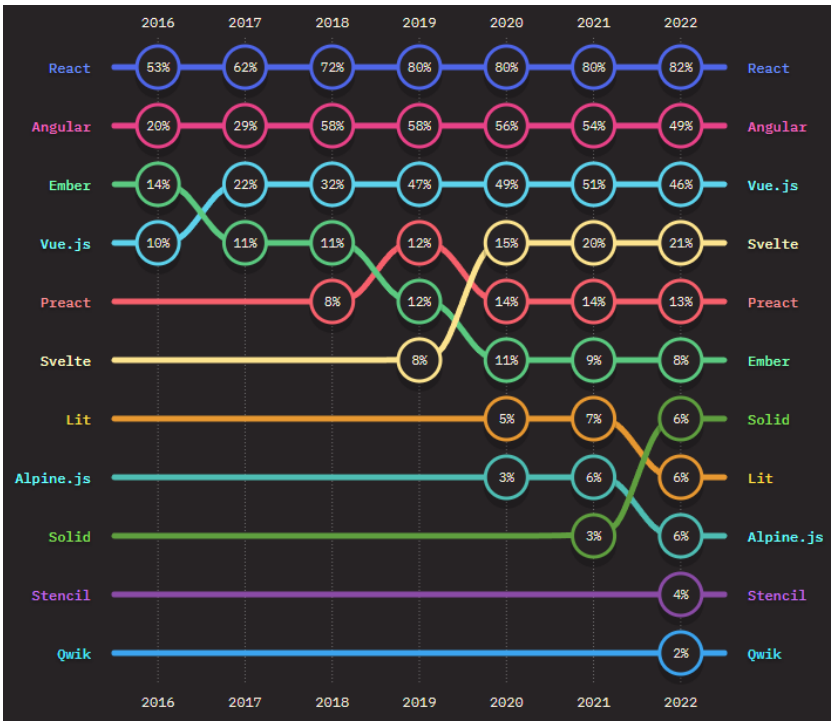
Prvním zjištěním z analýzy průzkumu State of JS 2022 je časová řada (viz Obrázek 4), na které je znázorněno, jakou úroveň povědomí mají vývojáři o jednotlivých frameworcích. Tato časová řada poskytuje pohled na to, jak se vnímání a povědomí o jednotlivých frameworcích vyvíjelo mezi vývojářskou komunitou napříč lety.



Obrázek 4 - Vývoj povědomí frameworků

zdroj: https://2022.stateofjs.com/en-US/libraries/front-end-frameworks/

Na první pohled je evidentní, že od roku 2017 do současnosti frameworky React, Angular a Vue.js dominují v podvědomí dotázaných vývojářů. Tyto technologie si upevňují svou pozici a zdá se, že zaujímají výsadní místo ve vědomí vývojářské komunity. Toto pozoruhodné povědomí může být důsledkem jejich rozsáhlého používání v průmyslu, kontinuální podpory a inovací, a také intenzivního zájmu a diskusí vývojářů o tyto konkrétní frameworky. Tímto způsobem se React, Angular a Vue.js staly klíčovými v oblasti webového vývoje a pevně zakotvily svou popularitu a význam ve světě moderního softwarového inženýrství.



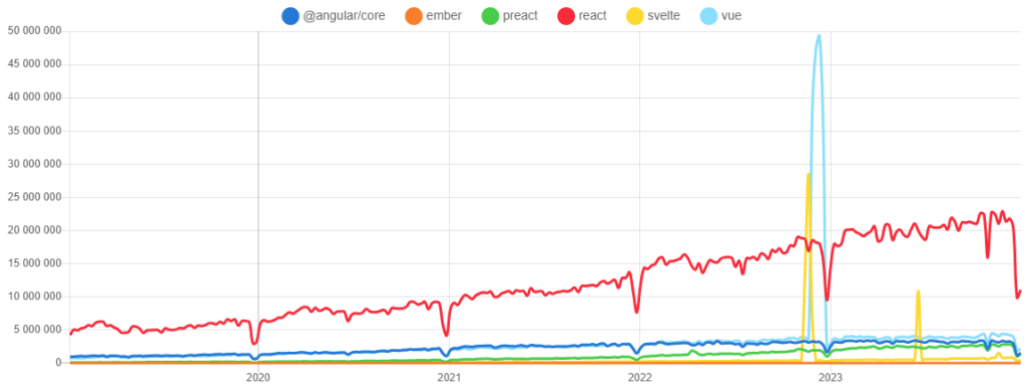
Obrázek 5 - Vývoj používanosti frameworků

zdroj: https://2022.stateofjs.com/en-US/libraries/front-end-frameworks/

Dalším zjištěním z tohoto průzkumu je časová řada (viz Obrázek 5), která zobrazuje používání frameworků vývojáři. Poskytuje dynamický obraz o tom, jak se preference a výběr frameworků vyvíjí v rámci vývojářské komunity.

Z obrázku je patrné, stejně jako u vývoje povědomí (viz Obrázek 4), že frameworky React, Angular a Vue.js si udržují vedoucí pozici v procentuálním užívání od roku 2017. Konkrétně lze vyčíst, že React zaznamenal významně vysoké procento užívání mezi dotázanými vývojáři, a to přibližně 82 %. Následně jsou zaznamenány významné hodnoty pro framework Angular, jehož užívání dosahuje přibližně 49 %, a pro Vue.js s procentuálním zastoupením kolem 46 %. Tato čísla podtrhují dominanci a přijetí těchto tří hlavních frameworků v komunitě vývojářů. Jejich vysoké procentuální zastoupení vypovídá o jejich širokém uplatnění a preferenci mezi profesionály v oblasti webového vývoje.

Posledním poznatkem je vizualizace z npm trends (viz Obrázek 6), která prezentuje vývoj počtu stažení frameworků v průběhu posledních pěti let. Na svislé ose je znázorněn týdenní počet stažení, zatímco na vodorovné ose je graficky znázorněn postup času. Tato grafická reprezentace poskytuje pohled na dynamiku stahování jednotlivých frameworků a umožňuje sledovat jejich vývojový trend v průběhu časového období.



Obrázek 6 - Vývoj počtu stažení

zdroj: https://npmtrends.com/@angular/core-vs-ember-vs-preact-vs-react-vs-svelte-vs-vue

Z analýzy grafu je zřejmé, že si React dlouhodobě udržuje pozici nejvíce stahovaného frameworku ve srovnání s ostatními frameworky. Totéž lez usoudit i z časové řady používanosti frameworků (viz Obrázek 5). Tato výrazná dominance Reactu v počtu stažení je patrná již několik let a naznačuje jeho trvalou přitažlivost vývojářské komunity. Graf dále ukazuje, že React je na vedoucí pozici i v průběhu času, představující konzistentní zájem a preferenci vývojářů při výběru frameworku pro své projekty. Dále je vidět, že ostatní frameworky jsou rozděleny do skupin, kde se v první skupině ustálily Angular, Vue.js a Preact. Ve druhé skupině se nachází Ember a Svelte.

Pozoruhodný jevem je také pokles stažení frameworků během období Vánoc a nového roku. Každoroční trend signalizuje, že během prázdnin dochází k dočasnému snížení aktivity v oblasti stažení frameworků. Naopak, ke konci roku 2022 došlo k prudkému nárůstu počtu stažení Vue.js a Svelte. To je přičítáno chybě v běhovém prostředí Deno. (20) (21) Tato situace ukazuje, jak externí faktory a události mohou ovlivnit dynamiku stažení frameworků a přinášet neočekávané pohyby v jejich popularitě. Ojedinělostí je nárůst počtu stažení Svelte v polovině roku 2023, která bylo nejspíš způsobeno vydáním verze Svelte 4. (22)

4.1.2 Vybrané frameworky

Výběr frameworků pro tuto analýzu byl založen na komplexním posouzení několika faktorů, včetně jejich umístění v relevantním průzkumu a celkového počtu stažení. Průzkum State of JS

poskytl cenný přehled o povědomí a preferencích vývojářů vůči různým frameworkům. Na základě tohoto průzkumu bylo zřejmé, že React, Angular a Vue.js zaujímají výsadní pozice v myslích vývojářské komunity.

Dalším důležitým faktorem byl celkový počet stažení, který poskytuje kvantitativní perspektivu na popularitu a aktuální užívání daných frameworků. React, s výrazným vedením v této statistice, ukázal na svou vysokou preferenci mezi vývojáři. Zatímco Angular a Vue.js, s nižšími hodnotami než React, stále demonstrovaly robustní základnu uživatelů, poukazující na jejich trvalou relevanci v oblasti webového vývoje.

Celkový výběr těchto tří frameworků – React, Angular a Vue.js – byl založen na kombinaci kvalitativních a kvantitativních dat, což umožňuje provést komplexní analýzu, která bude reflektovat aktuální stav a dynamiku ve světě webového vývoje.

	Verze	**Web**	**GitHub**	**Licence**	**Rok vydání**
React	18.2.0	react.dev	facebook/react	MIT	2013
Vue.js	3.3.13	vuejs.org	vuejs/core	MIT	2014
Angular	17.0.8	angular.dev	angular/angular	MIT	2016

Tabulka 1 - Přehled vybraných JS frameworků

4.2 Výběr kritérií ke zhodnocení

V předchozí kapitole byl proveden výběr frameworků, které budou podrobeny důkladné analýze a hodnocení. Nyní je nezbytné definovat jasná a objektivní kritéria, podle nichž budou tyto vybrané frameworky posuzovány. Stanovení těchto kritérií je důležité pro zajištění systematické a důkladné analýzy, která bude schopna objektivně posoudit flexibilitu, udržitelnost a další klíčové vlastnosti každého frameworku. Zároveň budou tyto kritéria sloužit jako pevný základ pro srovnávací analýzu, která přinese hlubší pochopení rozdílů a výhod mezi vybranými frameworky.

Výběr kritérií pro hodnocení frameworků, konkrétně *uživatelské zkušenosti*, *optimalizace pro prohlížeče, validace uživatelských dat* a *debugovací nástroje,* byl motivován zásadními oblastmi, které přímo ovlivňují vývoj webových aplikací a uživatelskou interakci. Uživatelské zkušenosti jsou zásadní, protože ovlivňují, jak uživatelé vnímají a interagují s aplikací. Optimalizace pro prohlížeče je důležitá vzhledem k rozmanitosti prohlížečů, u kterých rychlá odezva a efektivní vykreslování přispívá k celkové výkonové kvalitě webových aplikací. Validace uživatelských dat je podstatná z hlediska bezpečnosti a konzistence dat, což má vliv

na spolehlivost aplikace. Závěrem, debugovací nástroje jsou klíčové pro rychlou identifikaci a odstranění chyb, což zvyšuje efektivitu vývoje. Tyto aspekty byly vybrány s ohledem na jejich roli ve vývoji webových aplikací a schopnost každého frameworku přinést výhody v těchto klíčových oblastech.

4.2.1 Uživatelské zkušenosti

Hodnocení uživatelských zkušeností s frameworky bylo důležitým aspektem v procesu analýzy, posuzující efektivitu a vhodnost každého frameworku pro specifický vývojový kontext. Toto srovnání zahrnuje komplexní pohled na různé aspekty každého frameworku, včetně výkonnosti, flexibility, komunity a dostupných nástrojů. Důraz byl kladen na schopnost každého frameworku splnit specifické požadavky projektu a zároveň poskytnout vývojářům efektivní a uživatelsky přívětivé prostředí pro vytváření moderních webových aplikací. Celkové srovnání umožnilo identifikovat silné stránky každého frameworku a poskytne komplexní perspektivu na to, jaký nástroj nejlépe vyhovuje konkrétním potřebám vývojového týmu.

Hodnocení uživatelských zkušeností s frameworky bylo provedeno systematickým průzkumem dostupných zdrojů, které zahrnují oficiální dokumentaci jednotlivých frameworků, odborné recenze, výsledky průzkumů a šetření ve vývojářské komunitě. Analýza začala důkladným studiem oficiální dokumentace, kde byly identifikovány jednotlivé funkce, vlastnosti a doporučené postupy pro každý framework. Následně byly zohledněny odborné recenze a články, které poskytly nezávislý pohled na výhody a nevýhody každého frameworku.

Další důležitou součástí bylo sledování výsledků průzkumů a šetření ve vývojářské komunitě. Tato data poskytla reálný obraz o tom, jak jsou frameworky vnímány ve světě praktického vývoje, jakou mají podporu a jaké jsou jejich aktuální trendy. Zpětná vazba od komunity je klíčovým ukazatelem praktické použitelnosti a popularity každého frameworku.

Syntetizování nabytých informací o uživatelských zkušenostech umožnilo získat celkový obraz každého frameworku v souladu s definovanými kritérii. Tímto způsobem bylo zajištěno, že všechny aspekty frameworků byly pečlivě zváženy a hodnoceny na základě aktuálních informací a relevantních dat z různých zdrojů.

4.2.2 Optimalizace pro prohlížeče

Hodnocení optimalizace pro prohlížeče zahrnovalo analýzu schopností jednotlivých frameworků optimalizovat své výstupy pro různé prohlížeče a zařízení. Detailní průzkum bude

směřovat k posouzení kompatibility, optimalizace a způsobu implementace moderních technik pro zlepšení uživatelské zkušenosti. Taktéž byly zkoumány výkonové metriky, jako jsou rychlost načítání stránek a vykreslování obsahu, spotřeba paměti a další aspekty ovlivňující uživatelskou zkušenost. Celkové porovnání poskytlo všestranný pohled na to, jak jednotlivé frameworky pracují s optimalizací, která jsou důležitým aspektem pro vývoj moderních a interaktivních webových aplikací.

Hodnocení optimalizace pro prohlížeče bylo provedeno systematickým studiem dostupných zdrojů, ze kterých byl získán objektivní pohled na schopnosti každého vybraného webového frameworku v této oblasti. Prvním krokem byl průzkum oficiální dokumentace každého frameworku, která poskytla hlubší vhled do dostupných funkcí a doporučených postupů pro optimalizace webových aplikací. Dále byla provedena analýza odborných recenzí a článků od nezávislých expertů, kteří detailně posoudili praktické aspekty optimalizace v rámci každého frameworku. V potaz byly brány reálné zkušenosti vývojářů, zdůrazňující sílu a omezení optimalizace pro prohlížeče ve vývojovém prostředí.

V neposlední řadě byly zohledněna data z průzkumů a šetření ve vývojářské komunitě, která poskytla reálný obraz o tom, jak jsou jednotlivé frameworky vnímány a používány v praxi při optimalizaci webových aplikací pro prohlížeče. Kombinací těchto zdrojů bylo dosaženo souhrnného hodnocení schopností frameworků v oblasti optimalizace, které sloužilo jako základ pro informované rozhodnutí při výběru vhodného nástroje pro konkrétní vývoj projektu.

4.2.3 Validace uživatelských dat

V hodnocení byla provedena analýza vybraných webových frameworků validovat uživatelské vstupy. Tato fáze hodnocení se zaměřila na efektivnost implementace a správy procesu validace od získání uživatelských dat až po jejich zpracování. Bylo zkoumáno, do jaké míry frameworky nabízejí nástroje pro definici a provádění validací, včetně dynamické validace pro interaktivní formuláře. Byl kladen důraz na konzistenci, bezpečnost a uživatelsky přívětivý přístup k signalizaci chyb a poskytování nápovědy. Výsledky této analýzy poskytly pohled na to, jak každý framework zajišťuje správnost a bezpečnost uživatelských vstupů, což je nezbytné pro vytváření moderních a spolehlivých webových aplikací.

Hodnocení validace vstupů bylo provedeno průzkumem dostupných zdrojů s cílem získat souhrnnou a objektivní perspektivu na schopnosti každého vybraného webového frameworku v oblasti validace uživatelských vstupů. Prvním krokem bylo studium oficiální dokumentace

každého frameworku, kde byly identifikovány dostupné nástroje, metody a postupy pro implementaci a správu validace.

Dále byly vzaty v úvahu odborné recenze a články od expertů, kteří poskytli pohled na praktické stránky validace vstupů v rámci každého frameworku. Bylo sledováno, jak efektivně frameworky zvládají různorodé validace, včetně statických a dynamických situací, a jakou poskytují podporu pro správu chyb a nápovědu uživatelům. Data z průzkumů a šetření ve vývojářské komunitě byla rovněž důležitým zdrojem informací o praktickém využití a vnímáni frameworků při validaci vstupů. Výsledkem bylo komplexní a aktuální porovnání schopností jednotlivých frameworků v zajištění správnosti, bezpečnosti a přívětivého zpracování uživatelských vstupů.

4.2.4 Debugovací nástroje

Hodnocení debugovacích nástrojů důkladně zkoumalo schopnosti každého frameworku identifikovat a odstraňovat chyb během vývoje. Analyzovány byly nástroje poskytující ladící konzoli, sledování zásobníku a možnosti nastavování bodů přestávky. Důraz byl kladen na uživatelsky přívětivé prostředí a efektivní sledování proměnných v kódu. Kromě toho hodnocení zahrnuje podporu pro sledování výkonu a analýzu běhu aplikace v reálném čase.

Hodnocení debugovacích nástrojů bylo zpracováno prostřednictvím systematického průzkumu dostupných zdrojů. Prvním krokem byla analýza oficiální dokumentace každého vybraného frameworku za účelem identifikace, porozumění implementaci a funkcím jejich debugovacích nástrojů. Dále byly zohledněny odborné recenze od nezávislých expertů, kteří poskytli pohled na praktická hlediska a účinnost každého nástroje při řešení chyb a ladících úkolů. Zváženy byly také zkušenosti vývojářů sdílené ve vývojářských komunitách a diskusních fórech, což poskytlo užitečnou perspektivu z praxe.

Průzkum vývojářských blogů, článků a technických diskusí přispěl k podrobnému srovnání vlastností a výkonu jednotlivých nástrojů v různých vývojových scénářích. Kombinace těchto zdrojů umožnila sestavit informovaný přehled o přednostech a omezeních debugovacích nástrojů v kontextu každého webového frameworku, což následně poskytne užitečné informace pro vývojáře při volbě nejvhodnějšího prostředku pro úspěšné ladící aktivity.

4.3 Hodnocení frameworků

Tato kapitola představuje hodnocení a srovnání vybraných webových frameworků na základě předem definovaných aspektů a kritérií. Cílem bylo poskytnout užitečný pohled na jednotlivé frameworky, které umožní vývojářům informovaně volit nástroje pro své projekty. Hodnocení bylo provedeno metodou komparace a systematicky strukturováno kolem klíčových hledisek, jako jsou uživatelské zkušenosti, optimalizace pro prohlížeče, validace vstupů a schopnosti debugovacích nástrojů. Pro dosažení souhrnného srovnání byly zohledněny také praktické zkušenosti vývojářské komunity, oficiální dokumentace a nezávislé recenze. Tímto způsobem kapitole poskytuje podrobný a objektivní přehled o důležitých aspektech každého frameworku, což umožňuje vybrat nástroje odpovídající specifickým potřebám jejich projektu.

4.3.1 Uživatelské zkušenosti

React, Angular a Vue.js jsou tři populární frameworky pro vývoj webových aplikací. React je knihovna pro uživatelské rozhraní, Angular je plnohodnotný frontendový framework, zatímco Vue.js je progresivní framework, což znamená, že vývojář využívá pouze funkce frameworku, které potřebuje, a podle potřeby je rozšiřuje. Tyto frameworky lze téměř zaměnitelně použít pro vytváření frontendových aplikací, ale nejsou zcela stejné. Avšak každý z těchto frameworků využívá komponenty a umožňuje rychlé vytváření uživatelských rozhraní. (23)

Angular je subjektivní framework, který pracuje s technologiemi jako RxJS (rozšíření pro reaktivní programování (24)), pipes (používané v šablonových výrazech pro přijetí vstupní hodnoty a vrácení transformované hodnoty (25)), TypeScript a asynchronní programování. Jeho standardizovaný přístup k vývoji webových aplikací je důvodem, proč ho organizace preferují pro vývoj rozsáhlých aplikací. Nicméně tato jednoznačnost zároveň znamená strmější křivku učení. Zatímco většina frameworků vyžaduje znalost HTML, CSS a JavaScriptu, Angular vyžaduje porozumění hlavním konceptům a dalším technologiím, což může být náročnější pro začínající vývojáře. (27)

Flexibilita, výkon a jednoduchost použití učinily z Reactu oblíbenou volbu vývojářů po celém světě. Ačkoli někteří vývojáři mají smíšené pocity ohledně JSX (syntaktické rozšíření JavaScriptu, které umožňuje psát značky podobné HTML uvnitř JavaScript kódu (26)). Další výhodou Reactu je jeho Context API, což je vestavěné řešení správy stavu. Avšak jiná řešení pro správu stavu jsou výhodnější pro rozsáhlejší aplikace, Context API je stále skvělým nástrojem pro správu stavu v aplikacích postavených na Reactu. React dále nabízí nástroj

DevTools pro prohlížeč, který dále zjednodušuje vývojový zážitek DevTools poskytují užitečné informace o hierarchii komponent, změnách stavu a výkonnostních metrikách, které umožňují vývojářům identifikovat a efektivně řešit problémy. (27)

Vue.js se odlišuje tím, že nenutí vytvářet aplikaci od základu. Naopak umožňuje postupně přidávat Vue.js do konkrétních částí aplikace dle potřeby. Kromě toho má Vue.js nízkou křivku učení, čímž zjednodušuje zážitek z vývoje. Bez ohledu na to, zda jde o začátečníka nebo zkušeného vývojáře, může začít pracovat s Vue.js během několika minut, protože je postaven na základním HTML, CSS a JavaScriptu. Díky tomu je učení Vue.js snazší ve srovnání s jinými JavaScriptovými frameworky, které vyžadují znalost dalších značkovacích syntaxí a jazyku, jako jsou JSX a TypeScript. (27)

Ekosystém Angularu významně přispívá k jeho úspěchu díky silné podpoře komunity. Tento ekosystém poskytuje rozmanité zdroje, včetně blogových příspěvků, videí, ukázkových projektů a šablon, které zajišťují širokou dostupnost vzdělávacích materiálů pro vývojáře. V komunitě Angularu existuje bohatá paleta knihoven a nástrojů od třetích stran, které výrazně zlepšují vývojový proces. Tyto knihovny přinášejí doplňkové funkce, komponenty a nástroje, které mohou být integrovány do aplikací postavených na Angularu. Mezi ně se řadí například Angular Material poskytující UI komponenty nebo NgRx pro správu stavu. Z hlediska komponent nabízí Angular bohatou sadu oficiálních prvků v projektu Angular Material, který je podporován společností Google. Dále je třeba zdůraznit, že Angular je součástí akronymu MEAN (MongoDB, ExpressJS, Angular a NodeJS). Tato kombinace přináší výhodu toho, že je celá aplikace napsaná v jednom jazyce – JavaScriptu. (23) (27)

Reaktivní ekosystém kolem Reactu vyniká svoji silnou podporou open-source projektů a aktivní komunitou. Tento všestranný ekosystém poskytuje širokou škálu knihoven a nástrojů třetích stran, které rozšiřují funkcionalitu Reactu a usnadňují vývoje výkonných aplikací. Mezi klíčové součásti se řadí Redux a MobX pro správu stavu, React Router pro routování, Formik pro ověřování formulářů a CSS Modules pro efektivní stylování komponent. Dále React je základním kamenem pro vývoj výkonných frameworků jako jsou Next.js, Gatsby, Prereact, React Native a Remix, které rozšiřují možnosti vývoje a jsou silně podporovány komunitou. Díky své všestrannosti a rozmanitosti nástrojů se React stává atraktivní volbou pro vývojáře, ať už se jedná o webové či mobilní aplikace. React je součástí akronymu MERN (MongoDB, ExpressJS, React a NodeJS). Podobně jako u MEAN jsou frontend i backend plně postaveny na JavaScriptu, usnadňující jednotný vývoj na obou stranách aplikace. (23) (27)

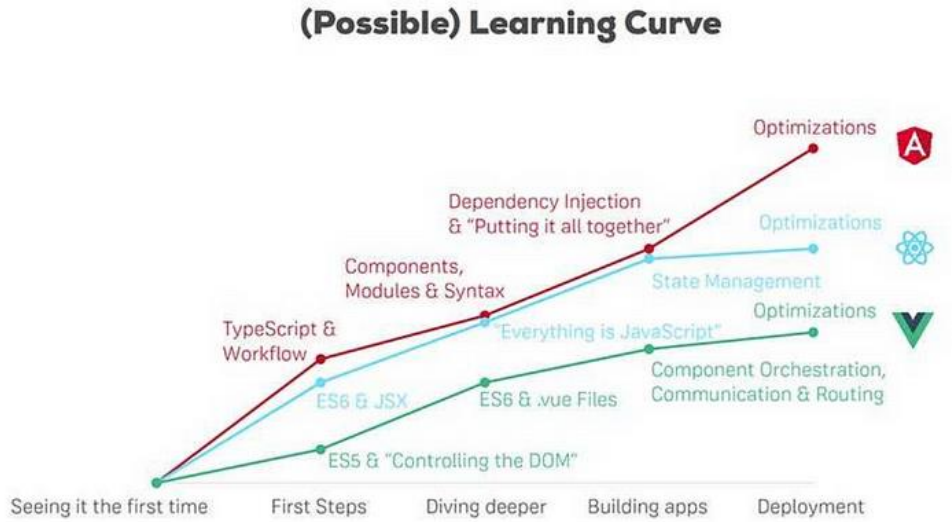
Ekosystém kolem Vue.js je aktivní a nabízí širokou paletu knihoven a pluginů třetích stran pro rozšíření jeho funkcionality. Tyto nástroje pokrývají škálu potřeb od správy stavu pro routování, což vývojářům usnadňuje nalezení řešení běžných problémů a zdokonalení svých pracovních postupů. Komunita kolem Vue.js poskytuje několik vlastních řešení a knihoven pro Vue.js 2 a Vue.js 3. Mezi užitečnými knihovnami v ekosystému Vue.js se nacházejí například Pinia pro správu stavu a VeeValidate pro ověřování formulářů. I když lze ve Vue.js využít Redux, neexistuje pro něj oficiální podpora. To však není překážkou, neboť Vuex je oficiální knihovna pro správu stavu speciálně navržena pro Vue.js aplikace. Kromě toho se dobře integruje s Vue.js, lze s ním také snadno ladit pomocí vývojářských nástrojů Vue.js. V počátcích Vue.js bylo obtížné najít hotové komponenty. S rozvojem komunity však nyní existuje široká škála vstupních komponent a pokročilých prvků, které vývojáři mohou využívat k urychlení svého vývoje. (23) (27)

Výhoda Angularu spočívá v důvěryhodnosti do budoucna díky podpoře od společnosti Google, zejména při vytváření rozsáhlých aplikací. Dále vynikající dokumentace Angularu představuje detailní průvodce jeho funkcionalitou. Každý koncept je podpořen konkrétními příklady ve srozumitelném jazyce, což umožňuje začátečníkům snadný přístup. Tato robustní dokumentace hraje roli v plynulém průběhu vývoje. Škálovatelnost Angular projektů při týmovém vývoji nabízí efektivní možnosti. I při drobných změnách provedených členem týmu není nutné aktualizovat celou strukturu projektu. Slabinou Angularu je učební křivka (viz Obrázek 7), vyžaduje totiž dovednosti v TypeScriptu. Z toho vyplývá, že učení Angularu vyžaduje více úsilí než u jiných frameworků. Také velikost projektu není ve prospěch Angularu, ovšem tento faktor je nejvíce citelný na malých projektech, u velkých projektů se tento rozdíl vyrovnává. (28)

K výhodám Reactu se řadí vytváření opakovatelných komponent, na které React klade důraz, čímž zjednodušuje programování a snižuje složitost kódu. Tento přístup nejen zvyšuje funkcionalitu, ale také přináší snadnou udržitelnost kódu. Další výhodou je využívání virtuálního DOM (Document Object Model), které přispívá ke zvýšení výkonu a zrychlení běhu aplikace. To činí React ideálním řešením pro tvorbu výkonných aplikací s rychlou odezvou. V neposlední řadě React vyniká svou přívětivou křivkou učení (viz Obrázek 7). Vývojáři webových aplikací se znalostí HTML a JavaScriptu si mohou rychle React osvojit, tím pádem je přístupným nástrojem i pro začínající vývojáře frontendu. Hlavní nevýhodou je tempo vývoje, které vede vývojáře k neustálému přeučování novějších koncepcí a udržování kroku s vývojem. Tato častá aktualizace pořád nutí vývojáře ke sledování novinek a přizpůsobování

se novým trendům. Přestože učení Reactu nevyžaduje žádné předchozí znalosti, má mnoho vývojářů problém s pochopením JSX. Kromě JSX si vývojáři také často stěžují na funkci online skriptingu v Reactu, která může být pro vývojáře obtížným úkolem. (28)

Vue.js vyniká svou přívětivou křivkou učení (viz Obrázek 7), díky čemuž je ideální volbou pro začínající vývojáře. Oproti Reactu a Angularu není zapotřebí předchozí znalosti TypeScriptu nebo pokročilého JavaScriptu. Velikost projektu vytvořeného ve Vue.js má přímý dopad na SEO (Search Engine Optimalization) webových stránek. Vyhledávače, včetně Google, preferují lehčí webové aplikace, aby zajistily rychlejší načítání. Schopnost Vue.js přizpůsobit se velikosti projektu představuje výhodu pro udržení vysokých standardů SEO a optimalizaci načítacích časů. Aktivní komunita a podpora fór jsou důležité pro úspěch každé technologie. Vue.js může počítat s živou komunitou a aktivními fóry, které poskytují užitečné zdroje pro učení a rychlé řešení případných problémů. Tato podpora přispívá k pozitivní křivce učení uživatelů a celkově zdůrazňuje přínosy používání Vue.js ve světě webového vývoje. Mezi omezení používání Vue.js se řadí ekosystém a to, jak se aplikace adaptují na různé prohlížeče a operační systémy. Vue.js má oproti Angularu a Reactu velmi omezený ekosystém, a proto nemusí správně fungovat ve starších verzích operačních systémů a webových prohlížečů. Dalším omezením Vue.js je jeho vývojářská komunita. Opět oproti Angularu a Reactu, které mají silnou podporu od Google a Facebook, nedisponuje takovou úrovní veřejné důvěry, která může ovlivnit přijetí Vue.js. (28)



Obrázek 7 - Srovnání křivek učení jednotlivých frameworků

zdroj: https://rahul-saini.medium.com/react-vs-angular-vs-vue-js-which-one-should-i-learn-in-2019-23ec05a49f78

React, Angular a Vue.js jsou tři populární frameworky pro vývoj webových aplikací, z nichž každý má své specifické vlastnosti. Angular nabízí plnou podporu od Google, zatímco React

vyniká flexibilitou a výkonem. Vue.js se odlišuje svou snadnou učenlivostí a postupným přístupem k vývoji. Každý z těchto frameworků poskytuje nástroje pro rychlé vytváření uživatelského rozhraní pomocí komponent. Ekosystém Angularu poskytuje širokou škálu zdrojů a nástrojů, včetně Angular Material a NgRx, což usnadňuje vývoj aplikací a zajišťuje robustní podporu. Reaktivní ekosystém Reactu nabízí širokou paletu knihoven a nástrojů, jako je Redux a React Router, které podporují vývoj výkonných aplikací pro různá prostředí. Vue.js má aktivní komunitu a široký výběr knihoven třetích stran jako je Pinia a VeeValidate, což zlepšuje produktivitu vývojářů. I když ve Vue.js lze využít Redux, oficiální podpora je poskytována pomocí Vuex, což zaručuje efektivní správu stavů aplikace. Rozvoj komunity Vue.js přinesl širokou škálu hotových komponent a pokročilých prvků, které urychlují vývoj webových aplikací. Angular nabízí důvěryhodnou podporu od Googlu a skvělou dokumentaci, ale učení je náročnější a projektům s menším rozsahem může chybět flexibilita. React usnadňuje tvorbu opakovatelných komponent a má přívětivou křivku učení, ale vyžaduje časté aktualizace a mnozí vývojáři zápasí s JSX, Vue.js nabízí snadné učení, přizpůsobení velikosti projektu a aktivní komunitu, ale jeho omezený ekosystém a nižší podpora od velkých společností mohou být pro některé vývojáře problém. Shrnutí této kapitoly se nachází v tabulce (viz Tabulka 2).

	Angular	**React**	**Vue.js**
Specifické vlastnosti	Plná podpora od Googlu, robustní dokumentace	Flexibilita, vytváření opakovatelných komponent, přívětivá křivka učení	Snadné učení, postupný přístup k vývoji, aktivní komunita
Nástroje pro vytváření UI	Široká škála zdrojů a nástrojů včetně Angular Material a NgRx	Široká paleta knihoven a nástrojů jako Redux nebo React Router	Široký výběr knihoven třetích stran jako Pinia a VeeValidate
Výhody	Rozsáhlá, dokumentace, robustní ekosystém	Flexibilita, velká komunita, široká paleta nástrojů	Snadné učení, aktivní komunita, široký výběr knihoven
Nevýhody	Náročnější učení, méně flexibilní pro menší	Časté aktualizace, problémy s JSX	Omezený ekosystém, nižší podpora od velkých společností

Tabulka 2 - Srovnání uživatelských zkušeností

4.3.2 Optimalizace pro prohlížeče

Optimalizace pro prohlížeče (SEO) hraje klíčovou roli při zajišťování toho, aby webové stránky nebo webové aplikace účinně dosáhly svých zamýšlených návštěvníků. Implementací nejlepších postupů SEO lze zvýšit viditelnost aplikací ve výsledcích vyhledávání, čímž přitahuje více organického provozu. Vylepšené pozice ve výsledcích vyhledávání vedou k vyšším mírám kliknutí, což má za následek zvýšenou angažovanost uživatelů a širší vystavení webové aplikace. (29)

Dále se SEO techniky nezaměřují pouze na zlepšení viditelnosti ve vyhledávačích, ale také na zlepšení celkové uživatelské zkušenosti. Optimalizace obsahu, navigace a struktury webových stránek může vytvořit uživatelský přívětivější prostředí, které podporuje spokojenost a oddanost návštěvníků. To zahrnuje zajištění rychlejšího načítání stránek, responzivitu na mobilních zařízeních a intuitivní navigaci, což vše přispívá k pozitivní uživatelské zkušenosti. (29)

SEO také umožňuje cílené marketingové úsilí tím, že dovoluje webovým aplikacím přizpůsobit svůj obsah a klíčová slova konkrétním demografickým skupinám. Optimalizace pro relevantní vyhledávací dotazy může přivést uživatele, kteří aktivně hledají produkty, služby nebo informace, které webová stránka nebo aplikace poskytuje. Kromě toho vysoké pozice ve výsledcích vyhledávání často korelují s důvěryhodností a spolehlivostí, protože uživatelé mají tendenci důvěřovat stránkám, které se ve výsledcích vyhledávání zobrazují mezi prvními. Proto upřednostnění SEO v aplikacích postavených na Reactu může nejen zvýšit viditelnost a uživatelskou zkušenost, ale také vytvořit důvěru a důvěryhodnost mezi cílovými uživateli. (29)

I když React umožňuje vytvářet statické, dynamické a jednostránkové aplikace, ty však nejsou na stejné úrovni, co se přívětivosti vůči SEO týče. Statické a dynamické webové stránky generují HTML soubory, které Google snadno indexuje a čte. Oproti tomu jednostránkové aplikace načítají pouze jednu HTML stránku a dynamicky mění tuto stránku v reakci na uživatele. U tohoto typu webové aplikace je server zodpovědný pouze za dodání první stránky HTML a veškerých potřebných dat. Webový prohlížeč klienta provádí veškerou logiku. V důsledku toho není potřeba každou akci aktualizovat a překreslit web, což vede k plynulé a reaktivní zkušenosti. Jednostránkové aplikace jsou závislé na souborech JavaScriptu, které pro SEO nejsou příliš užitečné, na rozdíl od statických a dynamických stránek, které generují soubory s informacemi v HTML, které jsou pro Google snadno čitelné. Problém je v tom, že

soubor HTML obsahuje pouze několik řádek kódu při přenosu zpět na stranu klienta. Proto, aby Google porozuměl obsahu webu a zaindexoval stránku, je tento kód nedostatečný. Google tedy musí počkat, až se obsah JavaScriptu stáhne, což může trvat nějakou dobu. Kvůli tomu mohou roboti Google okamžitě stránku opustit stránku, aniž by umožnili načtení obsahu a považovali ji za prázdnou. Existují způsoby, jak tomuto předejít:

- Předrendrování je efektivní technikou pro vytváření webových stránek přívětivých pro SEO, které indexační roboti mohou správně vyhodnotit. Jednoduše konvertuje JavaScriptové soubory na statické HTML, minimalizuje změny v kódu dobře doplňuje další online inovace. Nicméně má své nevýhody, jako je potřeba poplatku za služby a nevhodnost pro stránky s častými aktualizacemi dat.
- Izomorfní aplikace umožňují vývoj jak na straně serveru, tak na straně klienta. S izomorfním JavaScriptem lze pracovat s React aplikacemi a získat vykreslený HTML soubor, který mohou prohlížeče použít. To umožňuje indexovacím robotům, jako je Google, indexovat obsah. Izomorfní aplikace také zajišťují, že klientský prohlížeč může spustit skripty, ale v případě deaktivovaného JavaScriptu může obsah zobrazit i bez nich, což zlepšuje dostupnost a SEO. Vyvíjení real-time izomorfních aplikací může být obtížné, ale frameworky jako Gatsby nebo Next.js tento proces zjednodušují a urychlují. Gatsby kompiluje statické weby, zatímco Next.js je framework pro vývoj React aplikací s podporou vykreslování na straně serveru.
- Nejlepší způsob, jak zlepšit hodnocení při používání jednostránkových aplikací, je využít serverového renderování. Serverově renderované stránky jsou snadno indexovatelné a hodnotitelné roboty Google. Next.js je vhodný pro tvorbu server-side renderingu. Díky němu se JavaScriptové soubory převádějí na HTML a CSS, což umožňuje robotům Google získávat a zobrazovat data ve výsledcích vyhledávání podle potřeby. Tato metoda zajišťuje rychlou interakci uživatelů s webovou aplikací a optimalizaci pro sociální média, ale může zpomalit přechody mezi stránkami a zvýšit náklady na renderování na serveru. (30)

Výchozím stavem pro Angular aplikace je spouštění na straně klienta. To znamená, že první věcí, která se načte při spuštění aplikace, je prázdné HTML. Obsahuje pouze skript, který vykreslí veškerý obsah a naplní stránku. Prohlížeče a roboti procházejí texty a odkazy na stránce až po vykreslení JavaScriptu. To omezuje SEO potenciál jednostránkových aplikací v Angularu a vystavuje je potenciálním problémům. Ve verzích Angular 11 a novějších Google zahrnul

nové výchozí knihovny, aby byl Angular plně přívětivý k SEO. Tyto knihovny umožňují upravovat a nastavovat požadované meta značky, buď konfigurací Angular Universal pro spuštění v režimu předvykreslování nebo umožněním správy stavu aplikace samotné. (31)

Hlavní výzva spočívá v tom, jak roboti vyhledávačů interpretují obsah vykreslený pomocí JavaScriptu. Stejně jako u Reactu mají jednostránkové aplikace v Angularu mnoho obsahu dynamicky generovaného a vykreslovaného na straně klienta za pomoci JavaScriptu. I když Google v průběhu let výrazně zlepšil svou schopnost procházet a indexovat obsah JavaScriptu, proces není tak přímý jako u statického HTML. Google používá dvoufázový proces pro indexaci webových stránek s JavaScriptem. Nejprve prochází surový HTML dokument (fáze předvykreslení) a později, obvykle po několika sekundách až minutách, se vrací k vykreslení a indexaci obsahu generovaného pomocí JavaScriptu (fáze po vykreslení). Zpoždění mezi fázemi před vykreslením a po vykreslení znamená, že indexování JavaScriptem vykresleného obsahu může trvat déle. Toto zpoždění může mít potenciální dopad na viditelnost webové aplikace na stránkách s výsledky vyhledávání. (32)

Používáním sémantických HTML prvků jako <header>, <footer>, <article> a <section> poskytuje vyhledávačům důležitý kontext o obsahu stránky, což zlepšuje její indexovatelnost a optimalizuje využití indexačního rozpočtu. Avšak aplikace v Angularu často používají obecné prvky a pro vykreslování, které postrádají sémantický kontext tradičních HTML prvků. (32)

U jednostránkových aplikací v Angularu jsou různé pohledy obvykle načítány pod stejnou URL (Uniform Resource Locator) adresou, přičemž změny ve view jsou kontrolovány pomocí JavaScriptu. I když to poskytuje plynulý uživatelský zážitek, může to způsobit potíže se SEO. Vyhledávače mají potíže rozlišit tyto dynamicky načtené pohledy, protože jsou prezentovány pod stejnou URL adresou. To vede k nižšímu počtu indexovaných stránek a potenciální ztrátě organického provozu. (32)

Optimalizace Angular aplikace pro SEO vyžaduje podstatné plánování následované pečlivou implementací. I když Angular přináší některé výzvy v oblasti SEO, lze se správnými strategiemi výrazně zlepšit výkonnost SEO. Tato část se zabývá některými doporučenými postupy, pomocí kterých lze optimalizovat Angular aplikace. Je důležité si uvědomit, že SEO nejde jen o uspokojení vyhledávačů, ale také o zlepšení uživatelského zážitku. Pomocí těchto kroků lze optimalizovat Angular aplikace:

- Angular Universal je technologie, která umožňuje vykreslování Angular aplikací na straně serveru. To znamená, že aplikace může generovat statické HTML stránky na serveru, které mohou být odeslány místo surových JavaScript souborů. Tímto způsobem vyhledávače dostávají plně vykreslenou stránku, stejně jako by tomu bylo u statických stránek
- Pro menší aplikace nebo stránky, které neaktualizují obsah často, je předvykreslování alternativou k vykreslování na straně serveru. Předvykreslování generuje statické HTML stránky v době sestavení pro každou trasu aplikace, což usnadňuje vyhledávačům procházení a indexování. Nástroje jako Angular Pre-render se používají k vytvoření předvykreslených stránek pro Angular aplikace. Angular Pre-render prochází všechny trasy v aplikaci a každou z nich ukládá jako statický HTML soubor.
- HTML značky <title> a <meta> hrají klíčovou roli v SEO tím, že poskytují vyhledávačům informace o obsahu stránky. Pro dynamickou správu těchto prvků v Angular aplikaci lze použít služby Title a Meta poskytované balíčkem @angular/platform-browser.
- Lazy loading může výrazně zlepšit načítání a výkon Angular aplikací, což se následně pozitivně projeví na SEO. Umožňuje odložit načítání nekritického obsahu nebo obsahu pod dolní hranicí stránky, dokud není skutečně potřeba. Pokud jde o obrázky, lze využít direktivu NgOptimizedImage k implementaci techniky lazy loading a dalších optimalizačních metod. I když tato direktiva není přímo součásti Angularu, lze ji snadno integrovat. Tato direktiva řeší úkoly jako lazy loading, generování a vkládání responzivních obrázků a implementace moderních formátů obrázků jako WebP. (32)

Vue.js poskytuje významnou výhodu proti ostatním frameworkům, pokud jde o vykreslování webových komponent efektivní obnovování pohledu. Dosahuje toho tím, že aktualizuje pouze určené části stránky, místo aby opakovaně vykresloval celou stránku. To samozřejmě má pozitivní dopad na výkon. Nicméně Vue.js představuje značné výzvy pro optimalizaci pro vyhledávače. Standartně není SEO přívětivý a vyžaduje podstatné ladění. Vue.js nabízí deklarativní a komponentově orientovaný programovací model určený pro efektivní tvorbu složitých webových aplikací s jedinou stránkou. V případě těchto aplikací se typicky většina obsahu načítá dynamicky pomocí JavaScriptu. Kvůli tomu může značná část stránky vytvořené pomocí Vue.js zůstat skrytá pro vyhledávačové roboty. To vede k nižším hodnocením a nižší viditelnosti ve výsledcích vyhledávání. Aby se tento problém překonal a zajistilo se správné nastavení aplikací Vue.js pro SEO, musí vývojáři zohlednit SEO od samého začátku. Existuje

ovšem několik řešení, která jsou k dispozici k řešení souvisejících obav ohledně SEO a optimalizace Vue.js aplikací pro vyhledávače:

- Serverové vykreslování umožňuje generovat obsah webové stránky jako soubor HTML na serveru. Tento soubor je pak okamžitě k dispozici pro vyhledávače, což zlepšuje rychlost načítání a umožňuje rychlejší indexaci obsahu. Vue.js nabízí pro implementaci serverového vykreslování framework Nuxt.js.
- Pro generování plně vykreslených HTML stránek pro dynamický obsah lze využít předvykreslovací řešení, jako jsou například vue-prerender nebo ssr-vuejs-nodejs. Tento middleware zachytávají požadavky od vyhledávačů a předávají je do předrendrovacího enginu. Předrendrovací engine navštíví stránku, provede veškerý JavaScript, vyplní šablony obsahem a vrátí je zpět prohlížeči.
- Protože Vue.js je framework také pro jednostránkové aplikace, je obsažen v jediném souboru HTML. Každá další šablona je uložena v souborech .vue. Meta značky nemohou být uloženy v souborech .vue, protože přidáním nadpisů, meta popisků a kanonických URL generuje problémy s kódem. Meta značky hrají důležitou roli v Googleových hodnoceních. Přidání meta značek do frameworku je snadné za použití balíčku vue-meta.
- Vývojáři, kteří pracují s Nuxt.js, mohou využít modul nuxt-seo-kit vyvíjený Harlanem Wiltonem. Jedná se o všestranný SEO modul pro Nuxt 3, který umožňuje konfiguraci souboru robots.txt, HTTP hlaviček a meta značek. Také usnadňuje nastavení automatických kanonických URL. (33)

Jednostránkové aplikace poskytují dynamickou a interaktivní zkušenost pro uživatele, ale pro vyhledávače jako Google mohou být obtížně indexovatelné. Pro zlepšení SEO mohou vývojáři v Reactu využít techniky jako předrendrování, izomorfní vývoj nebo serverové renderování. Tyto postupy umožňují generování statického HTML obsahu, které je snadno indexovatelné prohlížečem. Frameworky Jako Gatsby nebo Nuxt.js poskytují nástroje pro implementaci těchto technik a zlepšují dostupnosti a indexovatelnosti jednostránkových aplikací. Výchozím stavem pro Angular aplikace je spouštění na straně klienta, což omezuje jejich SEO potenciál a vystavuje je potenciálním problémům. Verze Angular 11 a novější zahrnují knihovny pro plnou přívětivost k SEO, které umožňují úpravy meta značek a nastavení konfigurace pro spuštění v režimu předkreslování. I když Google zlepšil schopnost indexace obsahu JavaScriptu, proces indexace není tak přímý jako u statického HTML. Optimalizace

Angular aplikace pro SEO vyžaduje pečlivé plánování a implementaci, ale i správnými strategiemi lze výrazně zlepšit její výkonnost a viditelnost ve výsledcích vyhledávání. Vue.js poskytuje významnou výhodu v efektivním obnovování pohledu, aktualizuje pouze určité části stránky, což pozitivně ovlivňuje výkon. Nicméně optimalizace pro vyhledávače představuje výzvy, protože Vue.js není standardně přívětivý k SEO a vyžaduje podstatné ladění. Existují však řešení jako je serverové vykreslování pomocí frameworku Nuxt.js a předvykreslovací řešení, která zlepšují indexaci obsahu a viditelnost ve výsledcích vyhledávání. Pomocí balíčku vue-meta a modulu nuxt-seo-kit je možné snadno přidat meta značky a provádět další SEO optimalizace ve Vue.js aplikacích. Shrnutí této kapitoly se nachází v tabulce (viz Tabulka 3).

Framework	**SEO postupy**
React	Předvykreslování, izomorfní vývoj serverové renderování
Angular	Plná podpora od verze 11 pro úpravu meta značek, předvykreslování
Vue.js	Serverové vykreslování pomocí Nuxt.js, předvykreslovací řešení

Tabulka 3 - přehled SEO postupů

4.3.3 Validace uživatelských dat

Manipulace s uživatelskými vstupy je klíčem k tvorbě interaktivních webových aplikací v Reactu. Formuláře jsou velmi běžným a cenným nástrojem pro sběr informací od uživatelů. Mohou být použity pro autentizaci, registraci, zpětnou vazbu nebo pro sběr důležitých informací. V Reactu existuje velká komunita vývojářů, kteří vytvářejí knihovny pro důležité funkce jako je validace formulářů. Existuje mnoho knihoven pro snadnou validaci formulářů, takže vývojář nemusí psát veškerý kód sám. Komponenty Reactu často obsahují stavový objekt, který uchovává základní informace (data) pro reprezentaci nejnovějšího stavu komponenty, takže lze říci, že obsahuje nejnovější volby a vstupy uživatele. Dokumentace Reactu doporučuje ukládání vstupů uživatele do stavu, což je běžný JavaScriptový objekt. Lze tedy napsat JavaScriptovou funkci k ověření vstupů uživatele a případně zobrazit užitečné zpětné vazby, které pomohou uživatelům opravit své chyby. V Reactu jsou <input> komponenty, které jsou nazývané řízené a které získávají svou hodnotu ze stavu. Kdykoliv uživatel napíše nebo smaže něco z pole, událost onChange bude aktualizovat stav a pole pro vstup zobrazí aktualizovanou hodnotu. Vývojáři Reactu mohou používat Formik k validaci formulářů a zobrazování

užitečných chybových zpráv, pokud je to nutné. Yup je tvůrce schématu a velmi užitečný nástroj pro validaci formulářů. Lze jej použít k ověřování hodnot vstupů podle jejich typu, minimální a maximální délky, povinnosti, a dokonce určit vlastní chybové zprávy pro každý typ chyby. Formik umožňuje používat Yup stejně jako běžné JavaScriptové funkce pro validaci. Yup je nejužitečnější pro nastavování složité sady pravidel. Knihovna Formik sleduje stav formuláře a poskytuje vlastní komponenty pro snadnou validaci. <Formik> je hlavní vlastní komponenta, která dává přístup k metodám a hodnotám jako jsou aktuální hodnoty vstupů a chyby. Může být použita se standardními prvky <input> a <form>, stejně jako s vlastními komponentami Form, Field a ErrorMessage z knihovny Formik. (34)

Angular nabízí dva způsoby manipulace s validací formulářů: šablonově řízené a reaktivní formuláře. Obě metody využívají direktivy ke spojení validačních atributů HTML s validačními funkcemi frameworku. (35) Pro šablonově řízené formuláře poskytuje Angular několik vestavěných validačních nástrojů pro časté použití. K využití těchto nástrojů je nutné aplikovat validační atributy na každé pole formuláře, kde má být validace provedena. Tyto atributy jsou stejné jako běžné HTML5 validační atributy, jako je required, minlenght, maxlength atd. Na pozadí přiřazuje Angular tyto atributy k validačním funkcím definovaným ve frameworku. V reaktivních formulářích se nevyužívá direktiva ngModel ani HTML5 validační atributy. Validátory jsou specifikovány při vytváření objektů FormControl přímo v komponentě. Pro přidání vestavěných validačních funkcí pro FormControl lze předat ValidatorFn (funkce, která přijímá ovládací prvek a synchronně vrací mapu validačních chyb, pokud jsou přítomny, jinak je návratová hodnota nulová (36)). Na rozdíl od šablonově řízených formulářů se nepoužívají validační atributy. Místo toho lze předat příslušné ValidatorFn, například Validators.required, Validators.minlength(n) atd. (37)

Vue zjednodušuje manipulaci s formuláři pomocí direktivy v-model, která umožňuje dvousměrné vázání dat. To znamená, že změny vstupů formuláře automaticky aktualizují data komponenty a naopak. Pro jednoduchou validaci formulářů ve Vue.js jsou k dispozici vestavěné vlastnosti. S prvkem required lze určit povinná pole, zatímco atribut pattern umožňuje vytvářet vzory. To zjednodušuje proces validace na straně klienta. Pro podmíněnou validaci lze použít direktivy v-if nebo v-show k zobrazení nebo skrytí komponent formuláře v závislosti na určitých okolnostech. Tím lze aplikovat složitější validační logiku. S Vue.js je snadné validovat délku a složitost vstupních polí. Na základě uživatelského vstupu lze provádět dynamickou validaci pomocí výpočetních vlastností a metod. Vuelidate je známá validační knihovna pro Vue.js, která poskytuje sofistikované validační funkce. S Vuelidate je snadnější

manipulace s chybovými zprávami. Při porušení validačních pravidel lze rychle zobrazit příslušné chybové zprávy ve formuláři. S Vuelidate lze psát podmíněná validační pravidla závislá na stavu formuláře. Tato přizpůsobivost je zvláště užitečná pro složité formuláře, jejichž validační specifikace se mohou časem měnit. VeeValidate je validační knihovna pro Vue.js, která nabízí mnoho propracovaných validačních funkcí. S VeeValidate lze vytvářet pravidla validace přizpůsobené potřebám vývojáře. Tato adaptabilita je nezbytná pro správu konkrétních validačních požadavků. Díky podpoře internacionalizace (i18n) umožňuje VeeValidate zobrazovat validační zprávy v různých jazycích. VeeValidate umožňuje vytvářet dynamické formuláře tím, že vývojáři umožňuje přidávat a odebírat validační pravidla v reakci na uživatelský vstup. Tato vlastnost je zvláště užitečná pro dynamicky se měnící formuláře. (38)

Manipulace s uživatelskými vstupy je klíčová k tvorbě interaktivních webových aplikací v Reactu a formuláře jsou zde nezbytným nástrojem pro sběr užitečných informací od uživatelů. Velká komunita vývojářů Reactu vytvořila celou řadu knihoven, které usnadňují validaci formulářů a manipulaci s uživatelskými vstupy, jako je Formik. Komponenty Reactu často uchovávají stavový objekt, který zahrnuje nejnovější informace o volbách a vstupech uživatele, což umožňuje snadnou správu a aktualizaci formulářů v souladu s interakcemi uživatele. Použití knihovny Yup ve spojeni s Formik umožňuje efektivní a důkladnou validaci vstupů formulářů pomocí různých pravidel a vlastních chybových zpráv. Angular nabízí dva způsoby manipulace s validací formulářů: šablonově řízené a reaktivní formuláře, které jsou odlišné v použití validačních atributů a funkcí. Šablonově řízené formuláře využívají validační atributy HTML5, které se mapují na validační funkce frameworku Angular. Naopak u reaktivních formulářů se validační funkce specifikují přímo v komponentě pomocí objektů FormControl. Tyto funkce jako například Validators.required nebo Validators.minlength(n) jsou předány jako argumenty a zajišťují validaci bez použití validačních atributů. Vue.je zjednodušuje manipulaci s formuláři pomocí dvousměrné vazby dat díky direktivě v-model, která automaticky aktualizuje data komponenty podle změn vstupů. Pro jednoduchou validaci jsou ve Vue.js k dispozici vestavěné vlastnosti jako například required pro povinná pole a pattern pro vytváření vzorů. Podmíněná validace je možná pomocí direktiv v-if nebo v-show, které umožňují zobrazení nebo skrytí komponent formuláře v závislosti na okolnostech. Knihovny jako Vuelidate nebo VeeValidate poskytují sofistikované validační funkce a adaptibilitu pro různé potřeby vývojářů, včetně dynamických fomulářů a mezinárodní podpory. Shrnutí této kapitoly se nachází v tabulce (viz Tabulka 4).

Framework	Manipulace s formuláři	Validace formulářů	Knihovny
React	Komponenty Reactu uchovávají stavový objekt s informacemi o volbách a vstupech uživatele, což umožňuje snadnou správu formulářů	Použití Formiku a knihovny Yup umožňuje efektivní a důkladnou validaci vstupů formulářů pomocí různých pravidel a vlastních chybových zpráv	Formik, Yup
Angular	Šablonově řízené formuláře využívají validační atributy HTML5, zatímco u reaktivních formulářů se validační funkce specifikují přímo v komponentě	Validace formulářů je zajištěna pomocí vestavěných funkcí jako Validators.required nebo Validators.minlength(n), které jsou předány jako argumenty	@anugular/forms
Vue.js	Dvousměrná vazba dat umožňuje automatickou aktualizaci dat komponenty podle změn vstupů, což usnadňuje manipulaci s fomuláři	Vestavěné vlastnosti jako required a pattern usnadňují jednoduchou validaci formulářů, zatímco podmíněná validace je možná pomocí direktiv v-if nebo v-show	Vuelidate, VeeValidate

Tabulka 4 - přehled nástrojů pro validaci

4.3.4 Debugovací nástroje

Debugování je jednou z nejužitečnějších dovedností, kterou vývojář může ovládat. Pomáhá správně navigovat, odhalovat chyby v kódu a rychle a efektivně je opravovat, aby byly webové aplikace připravené k nasazení bez chyb a s vysokou kvalitou. V moderním průmyslu webového vývoje je to možné díky použití různých nástrojů a technik. (39)

React je jednou z nejrychleji rostoucích knihoven pro vývoj frontendu. Umožňuje vytvářet složité a interaktivní uživatelské rozhraní svým komponentově orientovaným přístupem. Stejně jako další vývojářské frameworky a knihovny pro vývoj uživatelského rozhraní má také React nástroje pro ladění. React Developer Tools je řešení pro prohlížení a analýzu komponent React. Lze s ním upravovat vlastnosti a stav komponent pro účely ladění. Také nabízí vestavěný profiler pro analýzu výkonu. Je k dispozici jako oficiální rozšíření pro prohlížeče a samostatná desktopová aplikace založená na Electronu. React Developer Tools rozšiřuje prostředí nativních vývojářských nástrojů prohlížeče o několik záložek pro ladění související s Reactem. Podobně jako samostatná aplikace nabízí React Developer Tools stejný zážitek z ladění jako rozšíření pro prohlížeč. Každý moderní webový prohlížeč nabízí vestavěné nástroje

pro vývojáře, aby si mohli prohlédnout a opravit chyby ve webových aplikacích. Například lze přidat body zastavení a použít konzoli k ladění libovolné aplikace založené na JavaScriptu. Také lze snadno prověřit nativní DOM prvky a atributy s nástroji pro vývojáře prohlížeče. React používá virtuální DOM s instancemi komponent a vykresluje skutečný dokument DOM během fází vykreslování komponenty, takže nelze použít vestavěné nástroje prohlížeče k ladění chyb specifických pro komponentu Reactu. React Developer Tools nabízí řešení pro ladění React aplikací tím, že umožňují manipulovat s vnitřkem komponent jako jsou vlastnosti a stav. Použití React Developer Tools tedy usnadní ladění aplikací React bez použití řešení zahrnující console.log() v prohlížeči. (39)

Angular DevTools je rozšíření prohlížeče pro Chrome a Firefox, navržené ke zlepšení schopností ladění a profilování Angular aplikací. Nejlépe funguje s Angular v12 nebo novější, pokud je aplikace zkompilována s vypnutou konfigurační možností optimalizace. Rozšíření DevTools se nachází pod záložkou Angular ve vývojářských nástrojích prohlížeče a po otevření zobrazuje dvě další záložky:

- Komponenty: Tato záložka umožňuje prozkoumat komponenty a direktivy aplikace, což umožní náhled nebo úpravy jejich stavů.
- Profiler: Tato záložka umožňuje profilovat aplikaci a pomáhá objevit výkonnostní slabiny během provádění detekce změn. (40)

Navíc zobrazuje verzi Angularu a nejnovější hash posledního commitu pro rozšíření. Angular DevTools pomáhá rychle prohlížet rozložení aplikace, zobrazuje strom komponent a direktiv. Lze zkontrolovat každou část, její podrobnosti a informace pomocí myši nebo klávesových zkratek. Záložka Profiler v Angular DevTools zaznamenává, jak Angular detekuje změny a poskytuje užitečné informace o událostech. Tato data lze zobrazit v jasném sloupcovém grafu, který zobrazuje každou část a její aktivitu během každého kroku. Profiler pomáhá vidět, jak Angular nalézá změny a poskytuje užitečné detaily o událostech. Zobrazuje jednoduchý sloupcový graf s aktivitou každé části během jednotlivých kroků. Obsahuje také zobrazení jako plamenový graf, který ukazuje umístění každé položky ve stromu zobrazení a čas použitý k nalezení změn. (40)

Při ladění Vue.js aplikací je běžné několikrát procházet kód, provádět změny, a poté restartovat server kvůli zjištění, zda problémy v aplikaci byly vyřešily. Přechod mezi zdrojovým kódem a prohlížečem však může být časově náročný proces. Pro zefektivnění ladění lze použít

rozšíření pro prohlížeče Vue.js Devtools. (41) Vue.js Devtools poskytuje různé užitečné nástroje, které pomáhají porozumět komponentám v aplikaci. Jednou z nejdůležitějších funkcí je zobrazení stromu komponent. Toto zobrazení se nachází v záložce Komponenty. Zde je hierarchický seznam komponent Vue.js, které jsou aktuálně aktivní v aplikaci nebo na stránce. Každá komponenta je reprezentována stromovým uzlem, který lze rozbalit a sbalit, aby mohl vývojář zjistit, jestli se komponenta vykresluje. Navíc zobrazení stromu komponent umožňuje interakci s komponentami v reálném čase. Například vývojář může kliknout na uzel komponenty, aby jej vybral, poté zobrazil a upravil její vlastnosti a data v panelu Devtools. Vue.js aplikace je obvykle složena z komponent, stavů a způsobů, jak jsou stavy sdíleny s komponentami. Občas se mohou objevit chyby nebo aplikace nemusí fungovat podle očekávání. S Devtools lze tyto fáze projít rychleji a snadno. Nejprve ve stromu komponent jsou zobrazeny stavy jednotlivých komponent. Jakékoliv změny dat se odrážejí ve vlastnostech komponenty v Devtools, čímž lze sledovat a provádět změny dat komponent v reálném čase. Dále se zde nachází inspektor časové osy. Inspektor časové osy umožňuje procházet předešlé verze stavů a událostí. S inspektorem časové osy lze diagnostikovat, zda události komponent vypadají podle očekávání. Při vytváření jednostránkových aplikací vývojář může snadno sledovat a ladit trasy a navigační proud. Záložka Routování zaznamenává data o trasách a historii během přechodů, navíc zobrazuje všechny trasy aplikace a jejich možnosti. (42)

React Developer Tools poskytuje prostředky pro prohlížení a analýzu komponent React, včetně možností úprav vlastností a stavu pro účely ladění. Tyto nástroje rozšiřují prostředí nativních vývojářských nástrojů prohlížeče a usnadňují práci s Reactem bez použití konzolových logů. Angular DevTools je užitečné rozšíření pro Chrome a Firefox, které zlepšuje schopnosti ladění a profilování Angular aplikací. Jeho záložky umožňují prozkoumat komponenty a direktivy aplikace a provádět profilování za účelem objevení výkonnostních slabých míst. Navíc zobrazuje důležité informace jako je verze Angularu a hash posledního commitu. Záložka Profiler zaznamenává, jak Angular detekuje změny a poskytuje užitečné informace o událostech, což napomáhá v optimalizaci výkonu aplikace.Vue.js Devtools je nenahraditelným nástrojem pro ladění Vue.js aplikací, který umožňuje rychlý přehled o stavu komponent a jejich hierarchii. S funkcí zobrazení stromu komponent lze jednoduše integrovat s komponentami v reálném čase a zjistit, jak jsou vykreslovány. Inspektor časové osy umožňuje procházet předešlé verze stavů a událostí, což je užitečné pro diagnostiku chyb a zjištění, zda komponenty fungují podle očekávání. Díky záložce Routování lze snadno sledovat navigační

proud a historii tras aplikace, což usnadňuje ladění jednostránkových aplikací. Shrnutí této kapitoly se nachází v tabulce (viz Tabulka 5).

Nástroj	Popis
React Developer Tools	Poskytuje prostředky pro prohlížení a analýzu komponent React, včetně možností úprav vlastností a stavu pro účely ladění. Rozšiřuje prostředí nativních vývojářských nástrojů prohlížeče.
Angular DevTools	Rozšíření pro Chrome a Firefox, které zlepšuje schopnosti ladění a profilování Angular aplikací. Obsahuje záložky pro zkoumání komponent a direktiv aplikace a profilování výkonnostních slabých míst.
Vue.js Devtools	Nenahraditelný nástroj pro ladění Vue.js aplikací, poskytuje přehled o stavu komponent a hierarchii. Umožňuje jednoduchou integraci s komponentami v reálném čase a zobrazení historie stavů a událostí

Tabulka 5 - přehled nástrojů pro ladění aplikací

5 Výsledky a diskuse

Kapitola je zaměřena na analýzu výsledků vlastní práce, která spočívala v posouzení vybraných frameworků na základě definovaných kritérií. V některých případech byly provedeny další průzkumy a sběr informací, které přispěly k hlubšímu porozumění rozdílů mezi jednotlivými variantami frameworků. Tyto dodatečné analýzy pomohly objasnit výsledky a zároveň poskytly cenné poznatky pro vyhodnocení frameworků z hlediska jejich vhodnosti pro konkrétní využití.

5.1 Analýza zhodnocení vybraných frameworků

V hodnocení uživatelských zkušeností se vybrané frameworky jevily jako vyrovnané (viz 4.3.1). Nicméně toto hodnocení je silně ovlivněno individuálními názory a zkušenostmi uživatelů s daným frameworkem. V oblasti optimalizace jednostránkových aplikací pro prohlížeče se však všechny tři frameworky potýkaly se stejnými problémy (viz 4.3.2). Pokud jde o validaci uživatelských dat, každý framework přináší své vlastní přístupy k řešení tohoto problému (viz 4.3.3). V neposlední řadě, všechny tři vybrané frameworky disponují výkonnými debugovacími nástroji s užitečnými funkcionalitami, které umožňují ladit různé typy chyb (viz 4.3.4).

5.1.1 Uživatelské zkušenosti

Angular s Reactem mají na rozdíl od Vue.js podporu od velkých společností, respektive Googlu a Facebooku. Tento vliv se podepisuje na velikosti komunit, kde jednoznačně vede React. Dále React vyniká flexibilitou a výkonem, přičemž křivka učení se nachází mezi Angularem a Vue.js (viz Obrázek 7). Díky své velké komunitě mohou vývojáři používat velkou škálu knihoven. Aby React ušetřil vývojářům čas, nabízí vytváření znovupoužitelných komponent. Velká komunita je tu také nevýhodou, protože udává rychlé tempo vývoje, a tak je nutné aplikace často aktualizovat. Začínající vývojáři mohou mít problémy se zápisem JSX.

Angular oproti oběma frameworkům poskytuje rozsáhlou dokumentaci. Ta v kombinaci s velkou komunitou nabízí široký výběr zdrojů a nástrojů, které mohou vývojáři využít. Díky tomu, že je Angular plnohodnotný UI framework, je jeho ekosystém velmi robustní, vyznačující se nejrůznějšími funkcionalitami. Také má ze všech tří zkoumaných frameworků nejstrmější křivku učení, zejména kvůli TypeScriptu, nadstavbou nad jazykem JavaScript, která jej rozšiřuje o mimo jiné statické typování známé z objektově orientovaných jazyků například C#

nebo Java. Kvůli své velikosti není vhodný na rozsahem menší projekty, kterým v takovém případě může chybět flexibilita.

Vue.js je ze zkoumaných frameworků nejsnazší na naučení (viz Obrázek 7), díky využívání jednoduchého HTML, CSS a JavaScriptu. Vue.js se také přizpůsobí velikosti projektu tím, že vývojář do projektu přidává funkce, které potřebuje. V porovnání s ostatními zkoumanými frameworky má nejmenší vývojářskou komunitu, zato velmi aktivní. Díky tomu mohou ostatní vývojáři vybírat z široké palety knihoven. Ovšem ekosystém má oproti Reactu a Angularu značně omezený, přičemž se mu dostává jen nízká podpora od velkých společností.

Dodatečně bylo zjištěno, že všechny frameworky jsou schopné pracovat s API (Application Programming Interface).

5.1.2 Optimalizace pro prohlížeče

Všechny zkoumané frameworky mají společný problém s optimalizací, a tím jsou jednostránkové aplikace. Také všechny frameworky mají jedno společné řešení, a to předvykreslování stránky. U Reactu je možnost izomorfního vývoje, kdy aplikace je spuštěna na straně serveru i klienta, nebo serverového vykreslení v momentě, kdy se klientovi vrací http odpověď. Angular nabízí knihovny pro úpravu meta značek. Vue.js stejně jako React umožňuje aplikaci vykreslit za pomocí serveru.

5.1.3 Validace uživatelských dat

Všechny frameworky mají knihovny, které zajišťují validaci formulářů. React k tomu využívá knihovny Formik a Yup umožňující důkladnou validaci. Angular obsahuje knihovnu @angular/forms, která obstarává validaci jak pro šablonově řízené, tak pro reaktivní formuláře. Vue.js nabízí knihovny Vuelidate a VeeValidate, které usnadňují jednoduchou validaci, ale také zvládají podmíněnou validaci.

5.1.4 Debugovací nástroje

React Developer Tools poskytují prostředky pro analýzu komponent včetně úprav vlastností a stavů, také rozšiřuje nativní funkce nástrojů v prohlížeči. Angular DevTools zlepšuje ladění a profilování aplikací, umožňuje prozkoumání komponent a direktiv a profilování výkonnostních slabých míst. Vue.js Devtools udržuje přehled o stavu komponent a hierarchii, umožňuje integraci v reálném čase a zobrazení historie stavů a událostí.

5.2 Analýza využití vybraných frameworků

Na základě získaných poznatků z praxe a dodatečného zkoumaní bylo provedeno zhodnocení možností využití vybraných frameworků. Perspektiva využití těchto frameworků může být posuzována z různých úhlů. Často se berou v potaz potřeby konkrétní aplikace a důležitost dostupnosti příslušných nástrojů a funkcionalit pro vývoj. Nicméně, frameworky nabízejí širokou škálu nástrojů a technik, což může vést k situaci, kdy více frameworků vyhovuje potřebám aplikace. Toto zhodnocení může zohlednit zkušenosti a preference vývojářů či vývojářského týmu.

React lze doporučit vývojářům, kteří preferují flexibilitu a výkon aplikace. Jeho jednoduché vytváření komponent umožňuje vývojářům volbu mezi funkcemi a třídami. Díky tomu, že v Reactu je vše vyjádřeno v JavaScriptu, je možné snadno pracovat se šablonami. React nabízí velkou a aktivní komunitu, což je pro mnoho vývojářů klíčovým faktorem. Vhodné typy aplikací, které lze pomocí Reactu vyvíjet, jsou jednostránkové aplikace, Instant Messaging aplikace, mobilní aplikace a aplikace pro produktivitu.

Angular lze doporučit vývojářům, kteří preferují rozmanité prostředí a minimalizují využití externích knihoven. Díky TypeScriptu je možné jej doporučit vývojářům, kteří upřednostňují striktně typovaný jazyk. Zároveň je vhodný pro vývojáře vyvíjející rozsáhlé aplikace, kterým nabízí řadu funkcionalit. Stejně jako u Reactu je jeho komunita četně zastoupena. Vhodné typy aplikací, které lze pomocí Angularu vyvíjet, jsou jednostránkové aplikace, podnikové webové aplikace, mobilní aplikace a desktopové aplikace.

Vue.js lze doporučit vývojářům, kteří s vývojem webových aplikací začínají. Díky jednoduchému HTML, CSS a JavaScriptu si vývojáři snadno a rychle osvojí používání tohoto frameworku. Umožňuje vytvářet komponenty, kde jsou bloky HTML, CSS a JavaScriptu v jednom souboru. V neposlední řadě je vhodný pro vývojáře, kteří pracují s menšími aplikacemi a funkcionalitu si mohou dle potřeby přidávat. Vhodné typy aplikací, které lze pomocí Vue.js vyvíjet, jsou jednostránkové aplikace, mobilní aplikace, progresivní webové aplikace a aplikace pro elektronické obchodování.

6 Závěr

Bakalářská práce byla tematicky zaměřena na problematiku vývoje webových aplikací pomocí JavaScriptových frameworků. Hlavním cílem práce bylo analyzovat a zhodnotit vybrané JavaScriptové frameworky vhodné pro vývoj webových aplikací. Dílčími cíli bylo charakterizovat problematiku vývoje webových aplikací a analyzovat možnosti využití frameworků pro tvorbu webových aplikací.

První část práce se zaměřila na charakteristiku vývoje webových aplikací a představení různých technologií spojených s vývojem v jazyce JavaScript. Byly popsány klíčové aspekty a náležitosti, které formulují prostředí vývoje webových aplikací. Tato část poskytla základní přehled o prostředí a aktuálních technologiích používaných při vývoji webových aplikací.

Během vlastní práce bylo nejprve na základě průzkumu State of JS 2022 (v době psaní této práce nebyly zpracovány výsledky z roku 2023) a npm trends vybráno šest potenciálních JavaScriptových frameworků pro hodnocení. Vzhledem k doporučenému rozsahu bakalářské práce byly pro samotné hodnocení vybrány pouze tři z nich. Výběr byl proveden na základě dostupných informací o jednotlivých frameworcích, včetně názorů vývojářů, které vycházely z výsledků State of JS 2022. Kritéria pro výběr zahrnovala *historii na trhu, aktivní používání frameworků v praxi a jejich osvědčení v reálných projektech, udržitelnost a aktivitu komunity* a *open-source povahu* frameworků. Dále bylo nutné sestavit kritéria pro hodnocení zvolených frameworků. Aspekty, pro výběr kritérií, zahrnovaly flexibilitu, udržitelnost, bezpečnost a další vlastnosti. Na základě těchto aspektů byla vybrána čtyři kritéria: *uživatelské zkušenosti*, *optimalizace pro prohlížeče*, *validace uživatelských dat* a *debugovací nástroje*.

Poslední část práce byla analýza možností využití vybraných frameworků k vývoji webových aplikací. Vzhledem k tomu, že hodnocené frameworky jsou vhodné pro podobné typy projektů, byla problematika posuzována jak z hlediska vývojářů, tak z hlediska typů aplikací, pro které jsou vhodné. Vývojářům, kteří preferují rychlost a flexibilitu je možné doporučit React. Angular je doporučen pro vývojáře, kteří preferují robustní funkcionalitu bez používání knihoven třetích stran. Vue.js lze doporučit vývojářům, kteří nemají velké zkušenosti nebo chtějí vyvíjet aplikace, které mají malou velikost.

7 Seznam použitých zdrojů

1. Team, Codecademy. What is a Web App? *Codecademy.* [Online] [Citace: 29. Červen 2023.] https://www.codecademy.com/article/what-is-a-web-app.

2. What is a Web Application? *Javatpoint.* [Online] [Citace: 29. Červen 2023.] https://www.javatpoint.com/web-application.

3. Web Development Technologies. *Orient Software.* [Online] [Citace: 4. Červenec 2023.] https://www.orientsoftware.com/technologies/web-technologies/.

4. Ellis, Danielle. What Web Developers Will be Prioritizing in 2023. *HubSpot blog.* [Online] 19. Leden 2023. [Citace: 4. Červenec 2023.] https://blog.hubspot.com/website/web-developers-priorities.

5. Web Application Security. *Synopsys.* [Online] [Citace: 5. Červenec 2023.] https://www.synopsys.com/glossary/what-is-web-application-security.html.

6. SQL Injection. *W3schools.* [Online] [Citace: 10. Červenec 2023.] https://www.w3schools.com/sql/sql_injection.asp.

7. HTML basics. *Mozzilla.* [Online] [Citace: 12. Červenec 2023.] https://developer.mozilla.org/en-US/docs/Learn/Getting_started_with_the_web/HTML_basics.

8. HTML Elements. *W3schools.* [Online] [Citace: 12. Červenec 2023.] https://www.w3schools.com/html/html_elements.asp.

9. Gasston, Peter. *Moderní web.* Brno : Computer Press, 2015. 978-80-251-4345-2.

10. CSS: Cascading Style Sheets. *Mozilla.* [Online] [Citace: 26. Červenec 2023.] https://developer.mozilla.org/en-US/docs/Web/CSS.

11. Rouse, Margaret. Cascading Style Sheets Level 3. *Technopedia.* [Online] 6. Únor 2014. [Citace: 27. Červenec 2023.]

12. A., Jordana. What Is JavaScript? A Basic Introduction to JS for Beginners. *Hostinger.* [Online] 1. Srpen 2023. [Citace: 2. Září 2023.] https://www.hostinger.com/tutorials/what-is-javascript.

13. Paruch, Zach. What Is JavaScript & What Is It Used For? A Basic Guide to JS. *Semrush Blog.* [Online] 21. Květen 2023. [Citace: 2. Září 2023.] https://www.semrush.com/blog/javascript/.

14. What Is React.js? A Look at the Popular JavaScript Library. *Kinsta.* [Online] 11. Červenec 2023. [Citace: 13. Září 2023.] https://kinsta.com/knowledgebase/what-is-react-js/.

15. Angular overview. *Sanity.* [Online] 15. Leden 2024. [Citace: 1. Březen 2024.] https://www.sanity.io/glossary/angular.

16. Staff, HubSpot. What Is Vue.js? (Uses + 5 Website Examples). *HubSpot.* [Online] 18. Červenec 2022. [Citace: 1. Březen 2024.]

17. Pavlik, Vlado. What Is SEO? Meaning, Examples & How to Optimize Your Site. *Semrush Blog.* [Online] 5. Prosinec 2022. [Citace: 2. Březen 2024.] https://www.semrush.com/blog/what-is-seo/.

18. Development, Web Application. How do you implement data validation and error handling for your web application's database input and output? *Linkedin.* [Online] 2. Březen 2024. [Citace: 2. Březen 2024.] https://www.linkedin.com/advice/0/how-do-you-implement-data-validation.

19. Lombardi, Phil. What Is Debugging? (Plus 8 Important Strategies To Try). *indeed.* [Online] 4. Únor 2023. [Citace: 3. Březen 2024.] https://www.indeed.com/career-advice/career-development/debugging.

20. marser_everlyn. What happened to Vue in Dec/2022? *Reddit.* [Online] 2022. [Citace: 26. Prosinec 2023.] https://www.reddit.com/r/vuejs/comments/101h1ul/what_happened_to_vue_in_dec2022/ .

21. Spare_Mix2041. What happend to npm downloads of vue? Seems there's some hack on downloads trend of vue for the last week. Or any event? *Reddit.* [Online] 2022. [Citace: 26. Prosinec 2023.] https://www.reddit.com/r/vuejs/comments/zd91ck/comment/iz0dh3v/?utm_source=share&utm_medium=web2x&context=3.

22. Announcing Svelte 4. *Svelte.dev.* [Online] 22. Červen 2023. [Citace: 26. Prosinec 2023.] https://svelte.dev/blog/svelte-4.

23. Pattakos, Aris. Angular vs React vs Vue 2023. *aThemes.* [Online] 9. Leden 2023. [Citace: 29. Prosinec 2023.] https://athemes.com/guides/angular-vs-react-vs-vue/.

24. The RxJS library. *Angular.* [Online] [Citace: 29. Prosinec 2023.] https://angular.io/guide/rx-library.

25. Transforming Data Using Pipes. *Angular.* [Online] [Citace: 29. Prosinec 2023.]

26. Writing Markup with JSX. *React.* [Online] [Citace: 29. Prosinec 2023.] https://react.dev/learn/writing-markup-with-jsx.

27. James, Nefe. Angular vs. React vs. Vue.js: Comparing performance. *LogRocket.* [Online] 31. Srpen 2023. [Citace: 29. Prosinec 2023.] https://blog.logrocket.com/angular-vs-react-vs-vue-js-comparing-performance/.

28. Joshi, Mohit. Angular vs React vs Vue: Core Differences. *BrowserStack.* [Online] 11. Květen 2023. [Citace: 30. Prosinec 2023.]

29. Onyekachukwu, Okosa Leonard. How to Make React Apps SEO-Friendly – A Handbook for Beginners. *freeCodeCamp.* [Online] 9. Leden 2024. [Citace: 30. Leden 2024.] https://www.freecodecamp.org/news/how-to-make-seo-friendly-react-

apps?fbclid=IwAR0_wC6e6cI3xbmV8NQvHYz9ZbMN6asV_CszP4Y3ObdwGKx9a5d_LGrosLE#importance-of-seo-in-your-react-applications.

30. Rose-Collins, Felix. Is React SEO-Friendly? React Search Engine Optimization Tips. *Tanktracker.* [Online] 4. Červenec 2023. [Citace: 30. Leden 2024.] https://www.ranktracker.com/blog/is-react-seo-friendly-react-search-engine-optimization-tips/?fbclid=IwAR2ZZ0flFQHBm3LHH8LxGEc66X69_ckhi3QJj5OPeLlpXuOmqkaR3IZ1PFI.

31. Kyslova, Kseniia. SEO tips and tricks for Angular web apps. *Proxify.* [Online] 24. Únor 2021. [Citace: 2. Únor 2024.] https://proxify.io/articles/is-angular-good-for-seo?fbclid=IwAR16Jd6EtrzwXefw9U0ZwlnKeqxfYHr82M-Sa4cUECwjQuKcZ8SmVCD5Gao.

32. Furgal, Aleksander. Angular SEO: 9 Tips and 5 Tools for Your Angular Website. *AsperBrothers.* [Online] 22. Červen 2023. [Citace: 5. Únor 2024.] https://asperbrothers.com/blog/angular-seo/?fbclid=IwAR0a8XAJXqymvSf6lhv-qCEsXwP1AQg5Y7e3QrjSwNDSyifw0oyB24L_zRo.

33. —. SEO in Vue.js With Vue-Meta, Vue Router, and Other Useful Tools. *AsperBrothers.* [Online] 13. Duben 2023. [Citace: 9. Únor 2024.]

34. Tchigladze, Irakli. Form Validation in React. *LeanyLabs.* [Online] [Citace: 10. Únor 2024.] https://leanylabs.com/blog/form-validation-in-react/.

35. Validating form input. *Angular.* [Online] [Citace: 11. Únor 2024.] https://angular.io/guide/form-validation.

36. ValidatorFn. *Angular.* [Online] [Citace: 11. Únor 2024.] https://angular.io/api/forms/ValidatorFn.

37. Ansari, Shadab. Angular Form Validation. *StackAbuse.* [Online] 21. Srpen 2023. [Citace: 11. Únor 2024.] https://stackabuse.com/angular-form-validation/.

38. Zia, Atif. Advanced Form Validation Techniques with Vue.js. *Medium.* [Online] 6. Listopad 2023. [Citace: 11. Únor 2024.] https://matifzia.medium.com/advanced-form-validation-techniques-with-vue-js-91731b6927ee.

39. Eze, Peter Ekene. Debug React apps with React Developer Tools. *LogRocket.* [Online] 9. Únor 2023. [Citace: 15. Únor 2024.] https://blog.logrocket.com/debug-react-apps-react-devtools/.

40. Paredes, Dany. How To Debug Angular Applications Easy. *Dev.* [Online] 17. červen 2023. [Citace: 18. Únor 2024.] https://dev.to/this-is-angular/simple-methods-for-debugging-angular-applications-1pe3.

41. Berning, Dave. How To Debug Components, State, and Events with Vue.js Devtools. *DigitalOcean.* [Online] 22. Leden 2022. [Citace: 20. Únor 2024.] https://www.digitalocean.com/community/tutorials/how-to-debug-components-state-and-events-with-vue-js-devtools.

42. Allotey, Charles. Debugging Magic with Vue Devtools. *Vue School.* [Online] 1. Srpen 2023. [Citace: 22. Únor 2024.] https://vueschool.io/articles/vuejs-tutorials/debugging-magic-with-vue-devtools/.

8 Seznam obrázků, tabulek, grafů a zkratek

8.1 Seznam obrázků

8.2 Seznam tabulek

8.3 Seznam použitých zkratek

Zkratka	Český název	Anglický název
HTML	Hypertextový značkovací jazyk	Hypertext Markup Language
CSS	Kaskádové styly	Cascading Style Sheets
UI	Uživatelské rozhraní	User Interface
API	Aplikační programové rozhraní	Application Programming Interface
JSX	JavaScript XML	JavaScript XML