Česká zemědělská univerzita v Praze

Technická fakulta

Bakalářská práce

Analýza a srovnání moderních jazyků pro tvorbu webových stránek

Vedoucí bakalářské práce: Ing. Zdeněk Votruba, Ph.D.
Autor práce: Aleksandr Kurbatov

ČESKÁ ZEMĚDĚLSKÁ UNIVERZITA V PRAZE

Technická fakulta

ZADÁNÍ BAKALÁŘSKÉ PRÁCE

Aleksandr Kurbatov

Informační a řídicí technika v agropotravinářském komplexu

Název práce

Analýza a srovnání moderních jazyků pro tvorbu webových stránek

Název anglicky

Analysis and comparison of modern languages for creating webpages

---

Cíle práce

Cílem práce je provézt revizi stávajících programovacích jazyků vhodných pro tvorbu webových stránek. Z těchto jazyků vybrat typické reprezentanty (min. 3) a tyto podrobně analyzovat z pohledu možného užití, rychlosti zpracování a dalších parametrů. Výsledkem práce bude souhrnné zhodnocení programovacích jazyků a to včetně předpokládaného vývoje.

Metodika

1. Analýza základních jazyků pro tvorbu web stránek
2. Explanace vybraných parametrů jednotlivých jazyků
3. Komparace jednotlivých jazyků ve vztahu s provedenu explanací
4. Indukce logiky jednotlivých jazyků
5. Syntéza zjištěných poznatků a jejich abstrakce
6. Volba parametrů testu
6. Matematické a statistické zpracování výsledků testování
7. Diskuse, průkaznost testů a závěru

Doporučený rozsah práce

30 až 40 stran textu včetně obrázků, grafů a tabulek

Klíčová slova

..

Doporučené zdroje informací

CASTRO, Elizabeth; HYSLOP, Bruce. *HTML5 a CSS3 : názorný průvodce tvorbou WWW stránek.* Brno: Computer Press, 2012. ISBN 978-80-251-3733-8.

GOODMAN, Danny. *Dynamic HTML : the definitive reference.* Sebastopol, CA: O'Reilly, 2007. ISBN 0596527403.

HAUSER, Marianne; HAUSER, Tobias; WENZ, Christian. *HTML a CSS : velká kniha řešení.* Brno: Computer Press, 2006. ISBN 80-251-1117-2.

ISAACS, Scott; DUDR, Jaroslav. *Dynamické HTML : vytváření interaktivních webových stránek s dynamickým obsahem, dynamickými styly, dynamickým polohováním.* Praha: Computer Press, 1998. ISBN 80-7226-083-9.

KOSEK, Jiří. *HTML : tvorba dokonalých www stránek : podrobný průvodce..*

LUBBERS, Peter; ALBERS, Brian; SALIM, Frank. *HTML5 : programujeme moderní webové aplikace.* Brno: Computer Press, 2011. ISBN 978-80-251-3539-6.

MIKLE, Pavol. *DHTML : dynamické HTML : referenční příručka.* Brno: Unis, 1997. ISBN 80-86097-09-9.

NIEDERST ROBBINS, Jennifer. *Learning Web design : a beginner's guide to (X)HTML, style sheets, and web graphics.* Sebastopol, CA ;: O'Reilly, 2007. ISBN 9780596527525.

SCHAFER, Steven M. *HTML, XHTML a CSS : bible [pro tvorbu WWW stránek] : 4. vydání.* Praha: Grada, 2009. ISBN 978-80-247-2850-6.

WEMPEN, Faithe. *HTML a CSS : krok za krokem.* Brno: Computer Press, 2007. ISBN 978-80-251-1505-3.

Předběžný termín obhajoby

2023/2024 LS – TF

Vedoucí práce

Ing. Zdeněk Votruba, Ph.D.

Garantující pracoviště

Katedra technologických zařízení staveb

Elektronicky schváleno dne 15. 2. 2022

doc. Ing. Jan Malaťák, Ph.D.

Vedoucí katedry

Elektronicky schváleno dne 23. 2. 2022

doc. Ing. Jiří Mašek, Ph.D.

Děkan

V Praze dne 24. 03. 2024

Čestné prohlášení

Prohlašuji, že jsem diplomovou/bakalářskou práci na téma: "Analýza a srovnání moderních jazyků pro tvorbu webových stránek" vypracoval/a samostatně a použil/a jen pramenů,které cituji a uvádím v seznamu použitých zdrojů.

Jsem si vědom/a, že odevzdáním bakalářské práce souhlasím s jejím zveřejněním dle zákona č. 111/1998 Sb., o vysokých školách a o změně a doplnění dalších zákonů, ve znění pozdějších předpisů, a to i bez ohledu na výsledek její obhajoby.

Jsem si vědom/a, že moje bakalářská práce bude uložena v elektronické podobě v univerzitní databázi a bude veřejně přístupná k nahlédnutí.

Jsem si vědom/a že, na moji bakalářskou práci se plně vztahuje zákon č. 121/2000 Sb., o právu autorském, o právech souvisejících s právem autorským a o změně některých zákonů, ve znění pozdějších předpisů, především ustanovení § 35 odst. 3 tohoto zákona, tj. o užití tohoto díla.

V Praze dne _____

Poděkování

Rád bych vyjádřil hluboké poděkování všem, kteří se podíleli na zpracování této bakalářské práce. Především bych chtěl upřímně poděkovat svému vedoucímu, Ing. Zdeňku Votrubovi, PhD, za jeho nepostradatelnou podporu, trpělivost a odborné vedení po celou dobu mého výzkumu. Jeho moudré rady a účast byly klíčové pro dosažení cílů této práce. Rád bych také poděkoval svým kolegům za cenné připomínky a pomoc při úpravě práce. V závěru bych chtěl vyjádřit hluboké poděkování své rodině a přítelkyni za jejich stálou podporu a inspiraci. Vaše laskavá slova, pochopení a pomoc byly během této cesty nezbytným zdrojem síly. Bez vaší pomoci a podpory by tento projekt nebyl možný.

Analýza a srovnání moderních jazyků pro tvorbu webových stránek

Abstrakt: Tato bakalářská práce je zaměřena na prostudování a určení nejvhodnějších z existujících nástrojů pro tvorbu webových stránek. Práce začne krátkým exkurzem do historie vzniku určitých programovacích jazyků pro tvorbu webových stránek. Tato kapitola a všechny následující budou rozděleny do dvou částí - jedna bude obsahovat analýzu front-endových jazyků - JavaScript, TypeScript, HTML, CSS a druhá se bude týkat back-endových jazyků - C#, Java, PHP, Golang, Python, Dart. Dále budou popsány hlavní výhody a nevýhody každého z těchto jazyků. Druhá část práce obsahuje praktický průzkum a určení favoritů v oblasti tvorby webových stránek - podle určitého souboru kritérií budou programovací jazyky mezi sebou vyhodnoceny a budou určeny ty nejlepší. Výsledkem práce je identifikace nejvhodnějších programovacích jazyků pro vývoj.

Klíčová slova: C#, Java, PHP, Golang, Python, Dart, JavaScript, TypeScript, HTML, CSS.

Analysis and comparison of modern languages for creating webpages

Summary: This bachelor's thesis aims to study and determine the most convenient existing tools for creating web pages. The work will begin with a brief overview of the history of the emergence of specific programming languages for creating web pages. This chapter and all subsequent ones will be divided into two parts - one will contain research regarding front-end languages - JavaScript, TypeScript, HTML, CSS, while the other will be about back-end languages - C#, Java, PHP, Golang, Python, Dart. The main advantages and disadvantages of each of the listed languages will then be described.

The second part of the thesis will contain practical research aimed at identifying favorites in the field of web development - based on a specific set of criteria, programming languages will be evaluated against each other, and the best ones will be identified. The result of the work is the determination of the most suitable programming languages for development.

Keywords: C#, Java, PHP, Golang, Python, Dart, JavaScript, TypeScript, HTML, CSS.

Obsah

1 Úvod

V době rychlé digitalizace a stále se vyvíjejícího online prostředí se tvorba webových stránek stala klíčovým prvkem pro efektivní komunikaci a prezentaci informací. Rozmanitost moderních jazyků pro vývoj webových stránek poskytuje vývojářům široký repertoár nástrojů, z nichž každý má své výhody a omezení.

Rychlý vývoj technologie a dynamický charakter internetového prostředí vyžadují neustálou aktualizaci znalostí v oblasti webového vývoje. V této práci budou zkoumány a porovnávány jazyky, jako jsou C#, ASP a .NET frameworky, Java, PHP. Dále budou zahrnuty modernější jazyky jako Golang, Python, Dart, JavaScript, TypeSript, a jejich specifické role při vytváření moderních webových stránek. Práce se bude zabývat také trendem používání frameworků a knihoven, které mohou vylepšit produktivitu vývojářů a standardizovat postupy. Vzhledem k tomu, že každý jazyk má své jedinečné vlastnosti, bude zdůrazněn výběr správné kombinace jazyků a nástrojů pro konkrétní vývojové úkoly. Důraz bude kladen na srovnání výkonnosti, škálovatelnosti, bezpečnosti a obecné flexibility těchto jazyků, aby bylo možné lépe porozumět, jakým způsobem splňují požadavky různých projektů.

Cílem této práce je poskytnout ucelený pohled na současnou problematiku v oblasti výběru jazyků pro webový vývoj a přispět k lepšímu porozumění klíčovým faktorům, které ovlivňují rozhodování vývojářů při výběru technologií pro své projekty.

2 Cíl práce a metodika

Cílem této bakalářské práce je analyzovat a porovnat existující nástroje pro tvorbu front-end a back-end webových stránek. Vybrat nejznámější programovací jazyky pro obě skupiny a porovnat je mezi sebou podle určitých kritérií - aktivita komunity, výkon, škálovatelnost a další parametry. Výsledkem práce bude vyhodnocení programovacích jazyků a určení od nejlepšího po nejhorší.

2.1 Metodika

Práce je rozdělena do několika částí. První část obsahuje úvodní materiál z hlediska historie každého z programovacích jazyků, které byly vybrány ke studiu. Další část obsahuje malé resumé výhod a nevýhod jednotlivých jazyků. Poté následuje druhá část, která je praktickou studií, a třetí částí je závěr, který je komplexním zhodnocením programovacích jazyků včetně jejich předpokládaného vývoje.

3 Teoretická část

3.1 Historie tvorby webových stránek

Historie webových stránek je zajímavá a trvá několik desetiletí a zahrnuje řadu klíčových milníků. Vznik webových stránek na internetu je poznamenán klíčovými momenty od samotného zrodu World Wide Webu. V roce 1989 vytvořil Tim Berners-Lee, inženýrský fyzik v CERN (Evropská organizace pro jaderný výzkum) koncept WWW (World Wide Web). V roce 1991 představil HTML (HyperText Markup Language), značkovací jazyk, který umožňoval strukturování a propojování dokumentů. Ten se stal výchozím bodem pro tvorbu webových stránek.

S vývojem jazyka HTML v polovině 90. let vznikly první prohlížeče, které uživatelům umožnily prohlížet webové stránky a pracovat s nimi. Prvními populárními prohlížeči byly Mosaic, uvedený v roce 1993, a Netscape Navigator, představený v roce 1994, které vzbudily široký zájem uživatelů.

V té době však byly webové stránky většinou statické a skládaly se z textu a obrázků. Významný průlom nastal s rozvojem dynamických technologií na konci 90. let, jako jsou programovací jazyky PHP a ASP na straně serveru. Ty umožnily vytvářet webové stránky, které mohly dynamicky reagovat na požadavky uživatelů, a poskytovat tak poutavější a interaktivnější zážitek.

Další etapa ve vývoji webových stránek přišla s rozšířením standardů CSS (Cascading Style Sheets) na přelomu 90. let 20. století a počátku 21. století. To umožnilo návrhářům oddělit vizuální návrh stránek od jejich struktury a poskytnout flexibilnější kontrolu nad stylem a rozvržením. Postupem času se s nástupem nových programovacích jazyků, frameworků a technologií staly webové stránky nejen prostředkem prezentace informací, ale také výkonným nástrojem pro vytváření interaktivních aplikací přizpůsobených různým zařízením a potřebám uživatelů.

Další důležitá etapa souvisí s vývojem dynamických webových aplikací v roce 2000. Technologie JavaScript a AJAX umožnily vytvářet interaktivní webové stránky, které aktualizují obsah, aniž by se celá stránka znovu načetla. To vedlo k výraznému zlepšení uživatelského komfortu a funkčnosti webových aplikací.

S nástupem éry mobilních zařízení v roce 2010 se stalo nutností přizpůsobit webové stránky různým obrazovkám. Standardem se stal adaptivní design využívající

CSS3 a HTML5, který poskytuje uživatelsky přívětivé prohlížení na mobilních zařízeních, tabletech a počítačích.

V posledních desetiletích se vývoj webových aplikací zaměřil na používání frameworků a knihoven, jako jsou React, Angular a Vue.js, což výrazně usnadnilo vytváření složitých rozhraní a zlepšilo výkon webových aplikací. V posledních letech se standardem staly progresivní webové aplikace (PWA), které poskytují možnost vytvářet webové aplikace, jež mohou běžet offline a poskytují vysokou úroveň uživatelského zážitku.

V současné době se webové technologie dále vyvíjejí, zavádějí nové standardy, jako je například WebAssembly, a poskytují vývojářům nové možnosti. Historie webu odráží úzký vztah mezi technologickými inovacemi a požadavky uživatelů a budoucnost této oblasti zůstává ovlivněna neustálým vývojem a zdokonalováním technologií virtuálního prostoru.

3.2 Programovací jazyky pro Front-End

Frontend je veřejná část webových aplikací (webových stránek), se kterou může uživatel přímo komunikovat a přicházet do styku. Frontend zahrnuje zobrazení funkčních úloh, uživatelské rozhraní, prováděné na straně klienta, a také zpracování uživatelských požadavků. Frontend je v podstatě vše, co uživatel vidí při otevření webové stránky.

3.2.1 JavaScript

V roce 1992 společnost Nombas začala vyvíjet vestavěný skriptovací jazyk s názvem C-minus-minus (Cmm). Cílem tohoto jazyka bylo vytvořit výkonný, ale snadno ovladatelný nástroj, který by mohl nahradit makra a být používán vývojáři. Společnost přejmenovala jazyk na ScriptEase a zabalila ho do produktu s názvem CEnvi, který představila vývojářům. Když popularita prohlížeče Netscape Navigator dosáhla vrcholu, Nombas vyvinula verzi CEnvi, kterou bylo možné používat na webových stránkách. Ta se stala prvním skriptovacím jazykem pro klienty používaným na internetu.

Se vzestupem popularity internetu a webových prohlížečů začala vznikat poptávka po klientních skriptovacích jazycích. Většina uživatelů internetu se připojovala prostřednictvím pomalých modemů, což ztěžovalo zpracování webových stránek. Společnost Netscape se rozhodla vyvinout klientní skriptovací jazyk, aby usnadnila

zpracování a snížila počet požadavků na server. Brendan Eich z Netscape zahájil vývoj a pojmenoval jazyk LiveScript. Před oficiálním vydáním byl však programovací jazyk přejmenován na JavaScript, aby využil popularitu názvu Java.

Microsoft, když viděla úspěch JavaScriptu, vydala vlastní verzi tohoto jazyka s názvem JScript. JavaScript a JScript se lišily ve své syntaxi a možnostech, což vytvářelo problém v oblasti absencí standardu. V roce 1997 byl JavaScript představen Evropskou asociací výrobců počítačů (ECMA) jako návrh pro standardizaci. Několik společností, včetně Netscape, Sun a Microsoft, se spojilo, aby vyvinulo standard ECMAScript. V následujícím roce byl tento standard přijat Mezinárodní organizací pro normalizaci a Mezinárodní elektrotechnickou komisí.

Od té doby různé webové prohlížeče implementují ECMAScript různými způsoby a používají ho jako základ pro své implementace JavaScriptu. To vše umožnilo JavaScriptu stát se nezbytnou součástí webového vývoje. (1)

3.2.2 TypeSript

TypeScript je programovací jazyk vytvořený uvnitř společnosti Microsoft, který byl zpřístupněn veřejnosti v roce 2012. Vývojem TypeScriptu byl pověřen Anders Hejlsberg, který byl také zodpovědný za vývoj C# a Turbo Pascal. TypeScript lze popsat jako "superset JavaScriptu" nebo "JavaScript s typy". Obsahuje existující syntaxi JavaScriptu a přidává novou syntaxi pro práci s typy. TypeScript zahrnuje program pro kontrolu typů, který analyzuje kód napsaný v JavaScriptu a TypeScriptu a hlásí chyby v typech. Kompilátor TypeScriptu spustí program pro kontrolu typů a vygeneruje ekvivalentní kód v JavaScriptu. Kromě toho TypeScript poskytuje jazykovou službu, která využívá kontrolu typů k poskytování užitečných nástrojů vývojářům v editorech, jako je například VS Code.

TypeScript zahrnuje čtyři věci:

Programovací jazyk

Jazyk, který zahrnuje veškerou existující syntaxi JavaScriptu, a navíc obsahuje novou syntaxi specifickou pro TypeScript pro definování a používání typů.

Kontrolor typů

Program, který přijímá sadu souborů napsaných v JavaScriptu a/nebo TypeScriptu, vyvíjí pochopení všech konstrukcí (proměnné, funkce...) vytvořených, a informuje nás, pokud si myslí, že je něco nastaveno nesprávně.

Kompilátor

Program, který spustí kontrolor typů, ohlásí případné problémy, a poté vytvoří ekvivalentní kód v JavaScriptu.

Jazyková služba

Program, který využívá kontrolor typů k informování editorů, jako je například VS Code, o tom, jak poskytnout užitečné nástroje vývojářům. (2)

3.2.3 HTML

Většina dnešních internetových technologií je založena na dlouho používaném jazyce HTML. Ten byl vyvinut pro značkování a návrh dokumentů umístěných na webových stránkách. Jazyk začal získávat své první vlastnosti v roce 1986. Podnětem bylo přijetí normy ISO-8879 Mezinárodní organizace pro normalizaci (ISO) - Standard Generalized Markup Language neboli ve zkrácené verzi SGML. K němu byl připojen popis, který uváděl, že SGML je určen pro strukturální značkování textu. Je pozoruhodné, že zde nebyl uveden žádný popis vzhledu dokumentu.

Z toho lze usuzovat, že SGML nebyl systém pro značkování textu a neměl žádný seznam strukturních prvků jazyka používaných v určitých souvislostech. Jazyk předpokládal popis syntaxe pro zápis hlavních značkovacích prvků. Po nějaké době dostaly název, dnes dobře známý - "značky". (11)

V souladu s tím byla zřejmá potřeba vytvořit jazyk, který by:

- Popisoval, jaký prvek je v jakých případech vhodné použít
- Obsahoval seznam prvků, které lze použít k vytvoření dokumentu čitelného různými programy.

Přestože se jazyk SGML, stejně jako jemu podobné aplikace, příliš nerozvíjel, nebyl zcela zapomenut. V roce 1991 oznámil Evropský institut pro fyziku částic potřebu

vyvinout mechanismus, který by umožnil přenos hypertextových informací prostřednictvím globální sítě. Právě SGML se stalo základem budoucího jazyka - Hyper Text Markup Language (HTML).(3)

3.2.4 CSS

CSS (Cascading Style Sheets) je jazyk stylů používaný pro tvorbu webových stránek. Vznik CSS souvisí s potřebou standardizovat design webových stránek a oddělit jej od struktury HTML a XML dokumentů. V roce 1994 Bert Boss zveřejnil článek, ve kterém představil myšlenku oddělení stylu a obsahu webových stránek. Tim Berners-Lee pak aktivně podporoval vývoj standardu, který byl realizován v rámci pracovní skupiny CSS v konsorciu W3C. První verze CSS (CSS1) byla představena v roce 1996 a obsahovala základní funkce pro návrh webových stránek. Následovala druhá verze (CSS2) v roce 1998 s přidáním dalších funkcí, jako je pozicování a tvorba tabulek. Verze CSS3 pak byla rozdělena do modulů a obsahovala pokročilé funkce, jako jsou animace, transformace a mediální dotazy. CSS zůstává důležitým nástrojem pro vývoj moderních webových stránek a poskytuje vývojářům kontrolu nad vzhledem obsahu.

3.3 Programovací jazyky pro Back-End

Backend - vytváření skriptů pro server, na kterém je umístěna webová aplikace (webová stránka).

Vývoj backendu zahrnuje tvorbu skriptů pro server, na kterém je web umístěn, vnitřní naplnění systému webových zdrojů, práci se serverovými technologiemi - návrh a vývoj programové logiky, interakci s databázemi (DB), práci s architekturou atd..

3.3.1 C#

Jazyk C# byl představen společností Microsoft v roce 2000 a stal se součástí .NET Frameworku. Společně s .NET Frameworkem byla představena technologie ASP.NET pro vytváření webových aplikací. Prvním modelem programování webových aplikací v jazyce C# pomocí ASP.NET byly webové formuláře, které poskytovaly abstrakce podobné Windows Forms. To zjednodušovalo vývoj vývojářům, zvyklým na vytváření desktopových aplikací.

V roce 2009 představila společnost Microsoft ASP.NET MVC, která nabízela flexibilnější a kontrolovanou architekturu. MVC umožňuje vývojářům rozdělit aplikaci na modely, pohledy a kontroléry, usnadňující testování a údržbu kódu. V roce 2016 byla vydána aktualizovaná verze ASP.NET Core, která je multiplatformní a s otevřeným zdrojovým kódem. Má vysokou výkonnost, nízkou hmotnost a podporu moderních technologií, a hodí se pro vytváření škálovatelných webových aplikací a mikroslužeb.

Jazyky C# a ASP.NET se neustále rozvíjejí a přidávají nové funkce a vylepšení, což umožňuje vývojářům psát efektivnější a čistý kód. Použití jazyka C# při vývoji webových aplikací je opodstatněno jeho jednoduchostí, výkonem a integrací s rozsáhlým ekosystémem technologií .NET. Celkově lze říci, že C# při vývoji webových aplikací má dlouhou historii, začíná s ASP.NET Web Forms a přechází k modernějším a flexibilnějším přístupům, jako jsou například ASP.NET MVC a ASP.NET Core.

3.3.2 Java

Java je všeobecně používaný, víceúlohový a objektově orientovaný programovací jazyk, vyvinutý ve společnosti Sun Microsystems v roce 1995. Jazyk byl původně navržen pro vestavěné systémy, ale koncem 90. let se stal široce využívaným v oblasti webového vývoje po vydání Java 2 Platform, Enterprise Edition (J2EE). S příchodem J2EE Java poskytla rozsáhlé knihovny a frameworky pro tvorbu škálovatelných a spolehlivých webových aplikací. Servlety a JavaServer Pages (JSP) se staly standardem pro tvorbu dynamických webových stránek, a technologie Enterprise JavaBeans (EJB) a Java EE byly používány pro vývoj velkoscale korporátních aplikací. Java získala popularitu ve vývoji webových aplikací díky frameworkům, jako jsou Apache Struts, Spring a Hibernate, a také ve světě mikroslužeb s příchodem frameworků a knihoven, například Spring Boot. Java si získala popularitu díky své platformní nezávislosti, spolehlivosti, výkonnosti a škálovatelnosti a zůstává jedním z předních programovacích jazyků pro vývoj serverové části webových aplikací. (8)(9)

3.3.3 PHP

PHP (PHP: Hypertext Preprocessor) je skriptovací programovací jazyk, který byl původně vytvořen pro vývoj webových stránek a dynamické programování. Začal jako sada skriptů v jazyce Perl v roce 1994, aby Rasmus Lerdorf mohl sledovat návštěvnost svého online životopisu. V roce 1995 byl vytvořen PHP/FI, který umožňoval vytváření

dynamických webových stránek včetně zpracování formulářů. V roce 1997 bylo jazyku PHP oficiálně dáno jméno "PHP: Hypertext Preprocessor" a začal získávat popularitu díky své jednoduchosti a snadnému použití. V roce 1998 byla vydána verze 3, která přidala podporu pro objektově orientované programování. Jazyk PHP se stal široce přijímaným nástrojem pro tvorbu dynamických webových stránek. Verze 4 z roku 2000 přidala další funkce a vylepšila výkon, včetně podpory objektově orientovaného programování. V roce 2004 byl vydán PHP 5 s novým Zend Engine 2.0, který přinesl další vylepšení výkonu a funkce. PHP 7 z roku 2015 přinesl významné zlepšení výkonu a nové funkce. PHP je v současnosti hojně používaný při vývoji webových stránek a aplikací, a to díky své integraci s HTML a funkcím pro práci s databázemi, zpracování formulářů, ověřování uživatelů a dalším úlohám.

3.3.4 Golang

Go neboli Golang je programovací jazyk vyvinutý společností Google v roce 2007. Hlavním cílem jazyka Go bylo vytvořit snadno použitelný jazyk, který by byl efektivní při vývoji a poskytoval vysoký výkon při provádění kódu. V roce 2009 byl jazyk Go veřejně představen a jeho vlastnostmi byly jednoduchá syntaxe, statické typování, efektivní správa paměti a inteligentní programování.

Od počátku jazyk Go poskytoval mnoho knihoven a balíčků pro vývoj webových aplikací, ale komunita aktivně experimentovala s různými frameworky a knihovnami. V roce 2010 se objevily první frameworky, Revel a Martini. Postupem času se však standardní knihovna Go vyvíjela a objevily se balíčky pro práci s webovými servery a zpracování požadavků HTTP, včetně balíčku net/http.

V následujících letech se objevily a staly se populárními různé frameworky pro vývoj webových aplikací v jazyce Go, například Gin, Echo a Fiber. Jazyk Go se také stal oblíbenou volbou pro implementaci architektury mikroslužeb díky svému výkonu, nízké abstrakci, podpoře konkurenceschopnosti a efektivní správě paměti. Jazyk Go se dnes hojně používá při vývoji webových aplikací k vytváření webových serverů, rozhraní API a mikroslužeb.

3.3.5 Python

Python je univerzální programovací jazyk, který se hojně používá při vývoji webových stránek. Vytvořil jej v roce 1991 Guido van Rossum s cílem vytvořit

jednoduchý a přehledný jazyk, který zvyšuje produktivitu vývojářů. Ve svých počátcích se Python používal k vytváření dynamických webových stránek pomocí rozhraní CGI. Poté se objevily první frameworky pro vývoj webových aplikací, například Zope, který poskytuje prostředí pro tvorbu webových aplikací s využitím OOP. V roce 2005 byl vydán framework Django, který se stal jedním z nejoblíbenějších nástrojů pro vývoj webových aplikací v jazyce Python. Flask se zase stal populárním mikroframeworkem pro tvorbu malých a středně velkých webových aplikací. V současné době nabízí Python mnoho frameworků a knihoven pro vývoj webových aplikací, například Pyramid, FastAPI, Tornado a další. Kromě toho vedl vývoj jazyka Python k vytvoření nástrojů, jako jsou například zápisníky Jupyter Notebook, které umožňují vytvářet interaktivní webové aplikace a vizualizace dat. Python je dnes jedním z nejoblíbenějších jazyků pro vývoj webových aplikací díky své jednoduchosti, čitelnosti kódu, rozsáhlému ekosystému a aktivní komunitě vývojářů.

3.3.6 Dart

Dart je programovací jazyk vytvořený společností Google, který se hojně používá k vývoji webových aplikací. Byl představen v roce 2011 a byl vyvinut jako alternativa k JavaScriptu pro tvorbu složitých webových aplikací. Dart obsahuje virtuální stroj Dart Virtual Machine (VM), který poskytuje vyšší výkon než JavaScript. Dart také podporuje kompilaci do jazyka JavaScript, což umožňuje jeho použití v moderních prohlížečích. Společnost Google interně používá Dart k vývoji webových aplikací, včetně frameworku AngularDart, který je založen na frameworku Angular. Koncem roku 2015 začala společnost Google aktivně vyvíjet framework Flutter pro mobilní a webové aplikace založený na Dartu. V roce 2019 byla představena experimentální verze Flutter for Web, která umožňuje vytvářet webové aplikace pomocí Dartu. Dart byl také standardizován jako jazyk ECMA, díky čemuž je lépe použitelný pro různé projekty. Celkově se Dart nadále vyvíjí a stále častěji se používá k vývoji webových aplikací, zejména v kontextu jazyka Flutter.

4 Metodologie testování

Cílem této kapitoly je popsat metodiku testování, která umožní srovnávací analýzu různých programovacích jazyků na základě předem stanovených kritérií. Před začátkem testování je nutné jasně formulovat a strukturovat kritéria, která budou použita pro hodnocení jednotlivých jazyků.

4.1 Kritéria hodnocení

4.1.1 Programovací jazyky back-endu

Výkon programovacího jazyka - kritérium výkonu programovacího jazyka hodnotí rychlost vykonávání kódu a efektivitu práce aplikací, které jsou s ním vyvíjeny. Tento aspekt je kritický při řešení úkolů, které vyžadují rychlé zpracování dat, škálovatelnost a optimalizaci algoritmů. Volba jazyka s vysokou výkonností zajišťuje stabilní a plynulý chod aplikací a umožňuje úsporu serverových prostředků a času vývojáře na optimalizaci. Takto efektivní využívání programovacích jazyků s vysokým výkonem se stává klíčovým prvkem úspěšné realizace projektů.

Aktivita komunity - Kritérium je důležité při výběru programovacího jazyka pro vývoj backendových aplikací. Analýza statistik dokončených projektů v různých jazycích poskytuje představu o zapojení vývojářů a popularitě daného jazyka v komunitě. Větší počet dokončených projektů naznačuje vyšší popularitu a rozšířenost jazyka, což znamená dostupnost zdrojů, knihoven, frameworků a podpory. Analýza statistik dokončených projektů je důležitým krokem při rozhodování o výběru jazyka pro vývoj backendových aplikací, protože pomáhá odhadnout úroveň aktivity a podpory ze strany vývojářské komunity.

Škálovatelnost programovacích jazyků - Škálovatelnost je klíčovým kritériem programovacích jazyků používaných k vývoji webových aplikací. Programovací jazyk musí být schopen efektivně zvládat rostoucí objem zátěže a zůstat produktivní i při rostoucí velikosti aplikace. Existují různé typy škálování, například vertikální škálování (zvyšování kapacity stávajících komponent) a horizontální škálování (přidávání nových instancí nebo uzlů pro rozložení zátěže). Programovací jazyky s dobrou škálovatelností poskytují vývojářům nástroje a řešení pro efektivní zvládání těchto typů škálování. Škálovatelnost je důležitá pro zajištění vysokého výkonu a dostupnosti webových aplikací i při zvyšujícím se počtu uživatelů a objemu dat. Nedostatečná škálovatelnost

může vést ke snížení výkonu, prodloužení doby odezvy a nedostupnosti aplikace, což negativně ovlivňuje uživatelský zážitek a obchodní ukazatele. Analýza škálovatelnosti různých programovacích jazyků je proto důležitá pro výběr nejvhodnějších nástrojů pro vývoj webových aplikací, které lze efektivně škálovat tak, aby splňovaly rostoucí potřeby a požadavky trhu.

Bezpečnost - kritériem bezpečnosti při výběru programovacího jazyka pro backendové aplikace je jeho schopnost chránit data a předcházet zranitelnostem. Důležitým aspektem je přítomnost a kvalita garbage collectoru, který automaticky spravuje paměť a předchází únikům paměti. Vysokoúrovňové jazyky, jako Java a Python, poskytují vyšší úroveň zabezpečení díky silnému typování, kontrole hranic polí a automatické správě paměti. Naopak nízkoúrovňové jazyky, jako C a C++, mohou být náchylnější k problémům se zabezpečením, ale poskytují větší kontrolu nad zdroji.

Testování a odstraňování chyb - jazyky s vestavěnou podporou automatizovaného testování, jako je Java a Python, umožňují vytváření a spouštění různých typů testů a automatizaci jejich analýzy. To zkracuje čas potřebný k testování a ladění aplikací a zvyšuje jejich spolehlivost a kvalitu. Jazyk Rust nabízí vestavěné testování ve standardní knihovně, což umožňuje rychlé psaní a spouštění základních testů bez potřeby instalace dalších nástrojů. Proto je důležité zvážit dostupnost testovacích nástrojů a vestavěnou podporu pro automatizované testování při výběru programovacího jazyka pro vývoj backendových aplikací.

4.1.2 Front-endové programovací jazyky

Výkon - při výběru programovacího jazyka pro vývoj backendu je důležité zohlednit kritérium výkonu, které se zaměřuje na efektivitu provádění kódu a schopnost zajistit vysokou rychlost provozu aplikace. Typizace proměnných je jedním z faktorů, které ovlivňují výkonnost. V jazycích se striktním typováním, jako je Java nebo C#, může překladač provádět statickou analýzu typů proměnných při kompilaci, což snižuje pravděpodobnost chyb za běhu a umožňuje optimalizaci kódu. Dynamicky typované jazyky, jako je Python nebo JavaScript, určují typy proměnných dynamicky za běhu, což může snížit výkon aplikace. Použití typizovaných proměnných ve vybraném programovacím jazyce může mít pozitivní dopad na výkon aplikace, protože snižuje chyby za běhu, zlepšuje optimalizaci kódu a snižuje režii kontroly datových typů. Toto je

důležitý aspekt při výběru programovacího jazyka pro backendový vývoj, zejména pokud je požadován vysoký výkon a efektivita.

Vlastnosti propojení s backendem - V kritériu se hodnotí pohodlí a efektivita připojení a interakce s backendem při výběru frontendového jazyka pro vývoj webových aplikací. Je důležité mít jazyk, který poskytuje pohodlné nástroje a knihovny pro práci s API rozhraním a výměnu dat s backendem. JavaScript, široce používaný jazyk pro vývoj frontendů, má bohatý ekosystém nástrojů, jako je Axios nebo Fetch API, které umožňují jednoduché a efektivní provádění HTTP požadavků na backend a zpracovávání dat. To umožňuje vytvářet dynamické webové aplikace, které komunikují se serverem bez nutnosti opětovného načítání stránky a poskytují lepší uživatelské rozhraní. U programovacích jazyků, jako je TypeScript nebo Dart, je také důležité zohlednit kompatibilitu s knihovnami a nástroji pro komunikaci s backendem pro efektivní propojení frontendové a backendové části aplikace. Při výběru frontendového jazyka je tedy důležité zvážit jeho schopnost pohodlné a efektivní komunikace s backendem a dostupnost vhodných nástrojů a knihoven pro práci s API rozhraním a výměnu dat.

Kompatibilita s prohlížeči - kritérium hodnotí schopnost jazyka vytvářet webové aplikace, které fungují správně v různých prohlížečích. JavaScript poskytuje širokou kompatibilitu díky standardizaci jazyka a jeho implementaci ve všech moderních prohlížečích. Nicméně, při vývoji složitých aplikací mohou existovat rozdíly v chování a podpoře funkcí mezi prohlížeči, a proto může být nutné použít polyfily nebo alternativní řešení. Preprocesory CSS, jako například Sass nebo Less, můžou také ovlivnit kompatibilitu stylů mezi prohlížeči, jelikož generují CSS, které musí být správně interpretovány všemi prohlížeči. Při výběru jazyka pro vývoj frontendů je důležité zvážit jeho schopnost zajistit kompatibilitu s prohlížeči a dostupnost nástrojů a přístupů k řešení problémů s kompatibilitou, aby bylo zajištěno dobré uživatelské prostředí v každém prohlížeči.

Aktivita komunity – kritérium hodnotí úroveň zapojení a podpory vývojářské komunity kolem daného jazyka. Aktivní komunita poskytuje zdroje, podporu a sdílené zkušenosti, což je důležité pro vývoj webových aplikací. Příkladem jazyka s velkou a živou komunitou je JavaScript, který disponuje velkým množstvím dokumentace, výukových zdrojů a vývojových knihoven. To umožňuje rychle nalézt odpovědi na otázky, sdílet znalosti a zapojit se do diskusí na fórech a sociálních sítích. Různorodost nástrojů a frameworků vyvíjených a podporovaných komunitou přispívá ke vzniku moderních a inovativních webových aplikací. Při výběru jazyka pro vývoj frontendů je

tedy důležité zohlednit aktivitu a podporu jeho komunity, protože to ovlivňuje úspěch a efektivitu projektu i profesní růst vývojáře.

5 Praktická část

5.1 Analýza programovacích jazyků

Předtím, než bude zahájeno porovnání programovacích jazyků, je důležité provést analýzu základních výhod a nevýhod každého z nich. To umožní získat objektivnější představu o tom, který jazyk nejlépe odpovídá požadavkům konkrétního projektu nebo úkolu. Znalost silných a slabých stránek jazyků pomáhá vývojářům přijímat fundovaná rozhodnutí při volbě nejvhodnějšího nástroje pro realizaci konkrétního úkolu. Tento přístup také přispívá k hlubšímu porozumění principům a vlastnostem každého jazyka, což zase zvyšuje efektivitu vývoje a kvalitu výsledného produktu.

5.1.1 Programovací jazyky pro Front-End

- **JavaScript**

Výhody:

Široké rozšíření: JavaScript je jedním z nejoblíbenějších programovacích jazyků, což znamená dostatečné množství zdrojů, knihoven a frameworků pro vývoj frontendu.

Množství frameworků a knihoven: Existuje mnoho frameworků a knihoven, jako například React, Angular, Vue.js, které zjednodušují vývoj frontendu poskytováním hotových řešení pro správu stavu, routování atd.

Rychlé provádění kódu na straně klienta: JavaScript se provádí na straně klienta, což umožňuje vytvářet dynamická a responzivní uživatelská rozhraní bez nutnosti odesílání požadavků na server a znovunačítání stránky.

Jednotný jazyk pro frontend a backend: JavaScript lze použít pro vývoj jak frontendu, tak backendu, což usnadňuje integraci a kompatibilitu mezi komponentami systému.

Aktivní komunita: JavaScript má velkou a aktivní komunitu vývojářů, což usnadňuje získávání podpory, řešení problémů a výměnu znalostí.

Nevýhody:

Křížová kompatibilita mezi prohlížeči: Různé prohlížeče mohou JavaScript interpretovat různě, což vyžaduje testování a dodatečný kód pro zajištění kompatibility s různými platformami.

Bezpečnost: JavaScript prováděný na straně klienta může být cílem útoků, jako jsou XSS (Cross-Site Scripting), pokud nejsou přijata vhodná bezpečnostní opatření.

Výkon: V rozsáhlých a složitých aplikacích se může JavaScript potýkat s výkonovými problémy kvůli nedostatečné optimalizaci nebo únikům paměti.

Asynchronnost a zpětná volání: Zpracování asynchronních operací a používání zpětných volání může vést ke složitému a chybovému kódu, zejména v případě hlubokého vnoření.

Nedostatek typování: JavaScript je jazyk s dynamickým typováním, což může vést k chybám za běhu kvůli neočekávaným datovým typům, zejména v rozsáhlých projektech.

- **TypeScript**

Výhody:

Statická typování: TypeScript poskytuje statické typování, což umožňuje odhalovat chyby v době kompilace a zvyšuje spolehlivost kódu.

Zlepšená čitelnost a podpora kódu: Díky statickému typování a možnosti používání rozhraní a datových typů je kód v TypeScriptu obvykle lépe čitelný a snáze udržovatelný.

Lepší integrace s IDE a vývojovými nástroji: Mnoho populárních integrovaných vývojových prostředí (IDE), jako je Visual Studio Code, nabízí rozšířenou podporu pro TypeScript, včetně automatického doplňování kódu, statické analýzy a rychlé navigace.

Velká komunita a ekosystém: TypeScript má aktivní komunitu vývojářů a rozsáhlý ekosystém knihoven a frameworků, což usnadňuje vývoj frontendu.

Podpora nejnovějších standardů JavaScriptu: TypeScript je obvykle aktualizován podle nejnovějších standardů JavaScriptu, což umožňuje využívat nové funkce jazyka a zároveň zachovávat výhody statického typování.(12)

Nevýhody:

Náročnost na výuku a implementaci: Zavedení TypeScriptu do projektu může vyžadovat čas a úsilí na jeho pochopení a adaptaci, zejména pro týmy zvyklé na práci s JavaScriptem.

Další kroky kompilace: TypeScript vyžaduje fázi kompilace do JavaScriptu před spuštěním kódu v prohlížeči, což může přidat mírnou dodatečnou zátěž při vývoji.

Nutnost definice datových typů: V TypeScriptu je nutné explicitně definovat datové typy pro proměnné, argumenty funkcí atd., což může být dodatečnou zátěží pro vývojáře.

Možnost potřeby aktualizace externích knihoven a frameworků: Některé externí knihovny a frameworky nemusí přímo podporovat TypeScript, což může vyžadovat dodatečnou práci při integraci.

Mírné zvětšení velikosti kódu: TypeScript může mírně zvětšit výsledný JavaScriptový kód kvůli zahrnutí informací o datových typech. (2)

- **HTML**

Výhody:

Snadné učení a používání: HTML je velmi jednoduchý a intuitivní jazyk, což ho činí přístupným i pro začátečníky vývojáře.

Široká podpora a standardizace: HTML je standardem pro tvorbu webových stránek a aplikací, a proto má širokou podporu od prohlížečů a různých vývojových nástrojů.

Flexibilita a rozšiřitelnost: HTML lze snadno rozšířit pomocí dalších jazyků značkování, jako je CSS pro styly a JavaScript pro dynamické chování, což poskytuje flexibilitu při vytváření různých uživatelských rozhraní.

Sémantické značkování: HTML poskytuje různé prvky pro vytváření sémanticky bohatých webových stránek, což zlepšuje dostupnost a SEO optimalizaci aplikací.

Nevýhody:

Omezené možnosti dynamického obsahu: HTML jako jazyk značkování má omezené možnosti pro vytváření dynamického a interaktivního obsahu bez použití dalších jazyků, jako je JavaScript.

Potřeba dalších jazyků a technologií: Pro tvorbu plnohodnotných webových aplikací na základě HTML je potřeba kombinace s dalšími jazyky a technologiemi, jako je CSS pro styly a JavaScript pro dynamické chování.

Složitost podpory velkých a složitých projektů: Ve velkých projektech může být HTML nevhodný pro podporu kvůli svým omezeným možnostem organizace kódu a správy komponent.

Závislost na struktuře stránky: Změna struktury HTML stránky může vyžadovat přezkoumání a úpravy v mnoha souborech a komponentech, zejména v rozsáhlých projektech.

Křížová kompatibilita prohlížečů: Některé prohlížeče mohou různě interpretovat a zobrazovat HTML kód, což může vést k nesrovnalostem ve vizuálním zobrazení a funkci aplikace na různých platformách. (11)

- **CSS**

Výhody:

Vylepšení vzhledu: CSS umožňuje stylovat webové stránky, čímž je činí atraktivnějšími a uživatelsky příjemnými.

Oddělení struktury a stylu: Použití CSS umožňuje oddělit strukturu HTML a vizuální vzhled elementů, což usnadňuje správu a změnu stylů bez zásahu do HTML kódu.

Responzivní design: CSS umožňuje vytvářet responzivní webové stránky, které se automaticky přizpůsobují různým velikostem obrazovek a zařízení.

Široká škála možností stylování: CSS poskytuje širokou škálu možností pro stylování, včetně barev, fontů, velikostí, rozmístění prvků a dalších.

Modularita a opakované použití: Styly CSS mohou být organizovány do modulů a opakovaně používány na různých stránkách a webových aplikacích, což snižuje duplikaci kódu a usnadňuje údržbu.

Nevýhody:

Složitost správy velkých projektů: Ve velkých projektech mohou být styly CSS složité a obtížně spravovatelné kvůli jejich globální povaze a dědičnosti.

Křížová kompatibilita prohlížečů: Různé prohlížeče mohou interpretovat CSS kód různě, což může vést k nesrovnalostem v zobrazení webových stránek a dodatečné práci na zajištění křížové kompatibility.

Omezené programovací možnosti: Na rozdíl od JavaScriptu má CSS omezené možnosti pro programování dynamického chování a interaktivity prvků.

Nedostatek podpory proměnných a funkcí: Do nedávné doby CSS nepodporoval proměnné a funkce, což mohlo komplikovat údržbu a aktualizaci stylů.

Omezené možnosti manipulace s prvky: CSS má omezené možnosti pro manipulaci s prvky a jejich chováním ve srovnání s JavaScriptem, což může omezit tvorbu složité funkcionality. (11)

5.1.2 Programovací jazyky pro Back-end

- **C#**

Výhody:

Výkon: C# je kompilovatelný jazyk s vysokým výkonem, což ho činí vhodným pro zpracování velkých datových objemů a aplikací s vysokou zátěží.

Bohatý ekosystém: C# má bohatý ekosystém nástrojů a frameworků, jako je .NET Framework a .NET Core, které poskytují mnoho hotových řešení pro vývoj webových aplikací.

Škálovatelnost: Díky výkonným nástrojům a frameworkům jsou aplikace v C# snadno škálovatelné, zvládají zvýšené zatížení bez výrazného poklesu výkonu.

Bezpečnost: C# má silný systém typů a možnosti omezení přístupu, což přispívá k vytváření bezpečných a spolehlivých webových aplikací.

Integrace s ostatními technologiemi: C# dobře spolupracuje s dalšími technologiemi, jako jsou databáze SQL Server, verzovací systémy, cloudové služby a další. (5)

Nevýhody:

Složitost učení: C# může být složitější k naučení, zejména pro začátečníky, ve srovnání s některými jinými programovacími jazyky.

Omezenost platformy: I když .NET Framework a .NET Core poskytují široké možnosti, jsou omezeny operačními systémy Windows, což může omezit výběr hostingu a nasazení.

Závislost na firemním ekosystému technologií: Použití C# pro backend může znamenat závislost na ekosystému Microsoftu, což může omezit výběr nástrojů a služeb.

Složitost nasazení na Linuxu: I když .NET Core podporuje nasazení na Linux, může to být složitější a vyžadovat více konfigurace, zejména pro týmy zvyklé na Windows.

Relativní novost v porovnání s některými jinými technologiemi: Ve srovnání s etablovanějšími technologiemi, jako je Java, může mít C# menší množství zdrojů a komunity pro podporu. (14)

- **Java**

Výhody:

Přenositelnost: Java má vysokou míru přenositelnosti, což umožňuje spouštění aplikací na různých operačních systémech bez změn v zdrojovém kódu.

Velká komunita: Java disponuje obrovskou a aktivní komunitou vývojářů, což zajišťuje přístup k množství zdrojů, knihoven a frameworků pro zjednodušení vývoje.

Škálovatelnost: Java má dobrou škálovatelnost, což umožňuje vytváření a škálování velkých a vysoce vytížených webových aplikací.

Spolehlivost: Java je stabilní a spolehlivý programovací jazyk, který je široce používán v korporátním prostředí pro tvorbu kriticky důležitých systémů.

Bezpečnost: Java nabízí mnoho vestavěných bezpečnostních mechanismů, jako je Java Virtual Machine (JVM), která poskytuje kontrolu nad spouštěním kódu a chrání aplikace před mnoha druhy útoků. (13)

Nevýhody:

Velké množství kódu: Vývoj v Javě často vyžaduje více kódu k dosažení stejných výsledků než některé jiné jazyky, což může zpomalit proces vývoje.

Složitost učení: Java je relativně složitý programovací jazyk, zejména pro začínající vývojáře, a vyžaduje čas na naučení.

Vysoké paměťové nároky: Java Virtual Machine (JVM) vyžaduje značné množství operační paměti, což může vést k většímu využití paměti na serveru.

Pomalejší startovní rychlost aplikace: Spuštění webových aplikací v Javě může trvat déle kvůli inicializaci JVM a dalších komponent.

Méně vhodné pro mikroslužby: Java může být méně vhodná pro vytváření mikroslužeb kvůli svému velkému rozsahu a složitosti ve srovnání s některými lehčími jazyky, jako je Go nebo Node.js.

- **PHP**

Výhody:

Jednoduchost učení: PHP je relativně snadný jazyk pro začátečníky, což usnadňuje učení a rychlý start vývoje.

Široká dostupnost hostingových služeb: Mnoho poskytovatelů webhostingu podporuje PHP, což umožňuje snadné nasazení webových aplikací.

Bohatá dokumentace a komunita: PHP má bohatou dokumentaci a aktivní komunitu vývojářů, což usnadňuje získání podpory a řešení problémů.

Flexibilita: PHP je velmi flexibilní jazyk, který umožňuje různé přístupy k vývoji a integraci s různými technologiemi.

Velké množství dostupných knihoven a frameworků: Existuje mnoho knihoven a frameworků pro PHP, které usnadňují vývoj a poskytují hotová řešení pro běžné úkoly.

Nevýhody:

Omezená škálovatelnost: PHP není tak škálovatelný jako některé jiné jazyky, což může vést k problémům s výkonem a škálovatelností při vývoji velkých a složitých aplikací.

Nedostatečná modularita: PHP trpí nedostatkem modularity ve srovnání s některými modernějšími jazyky, což může vést k obtížím při správě a údržbě kódu v rozsáhlých projektech.

Bezpečnostní otázky: PHP může být náchylný k bezpečnostním zranitelnostem, zejména pokud není správně konfigurován a používán.

Méně efektivní pro komplexní aplikace: Při vývoji velkých a složitých aplikací PHP může být méně efektivní než některé jiné jazyky, jako je například Java nebo C#.

Nízká výkonová efektivita: PHP může mít nižší výkonovou efektivitu ve srovnání s některými jinými jazyky, což může vést k pomalejší odezvě a zátěži serveru při zpracování požadavků.

- **Golang**

Výhody:

Vysoký výkon: Go je kompilovaný jazyk s efektivní správou paměti, což zajišťuje vysoký výkon při provádění.

Jednoduchost a čistota kódu: Go nabízí jednoduchou a srozumitelnou syntaxi, což usnadňuje psaní čistého a snadno udržovatelného kódu.

Konkurenceschopnost a paralelismus: Vestavěná podpora goroutines a channels usnadňuje vývoj paralelních a konkurenčních aplikací v Go.

Škálovatelnost: Go je snadno škálovatelný pro zpracování velkých objemů dat a vysokých zátěží díky své efektivní práci s paralelismem a konkurencí.

Bohatá standardní knihovna: Go je dodáván s rozsáhlou standardní knihovnou, která obsahuje mnoho užitečných balíčků pro vývoj webových aplikací. (4)

Nevýhody:

Relativní novost: Go je poměrně novým jazykem programování a jeho ekosystém může být méně rozvinutý a méně obsahovat třetích stran knihoven a nástrojů ve srovnání s jinými zavedenými jazyky.

Omezená podpora některých technologií: Některé technologie a platformy mohou mít menší podporu v Go, což může způsobit problémy při integraci s existujícími systémy.

Složitost asynchronního programování: I když Go podporuje konkurenční programování, asynchronní programování v Go může být pro některé vývojáře složité kvůli specifikám jeho modelu provádění.

Omezené možnosti nastavení serveru: Go poskytuje základní možnosti pro nastavení webového serveru, ale v některých případech mohou chybět pokročilejší funkce, které poskytují jiné serverové jazyky.

Nedostatek nástrojů a balíčků pro některé úkoly: Přestože má Go bohatou standardní knihovnu, pro některé úkoly může být potřeba použít třetí strany balíčky nebo nástroje, které mohou být méně běžné v komunitě Go. (4)

- **Python**

Výhody:

Jednoduchost a čitelnost kódu: Python je známý svou jednoduchostí a čitelností kódu, což usnadňuje vývoj a údržbu aplikací.

Rozsáhlá standardní knihovna: Python je dodáván s bohatou standardní knihovnou, která obsahuje mnoho užitečných modulů pro vývoj webových aplikací bez potřeby instalace dalších třetích stran.

Množství frameworků: Python nabízí mnoho populárních frameworků, jako jsou Django, Flask, Pyramid, které usnadňují tvorbu webových aplikací poskytováním hotových řešení pro zpracování požadavků, routování a práci s databázemi.

Široká podpora a komunita: Python má obrovskou a aktivní komunitu vývojářů, která poskytuje mnoho zdrojů, knihoven a fór pro získání podpory a řešení problémů.

Multiplatformnost: Python je multiplatformní jazyk, což znamená, že aplikace vytvořené v Pythonu mohou běžet na různých operačních systémech bez úprav. (7)

Nevýhody:

Nízký výkon: Python může být pomalejší ve srovnání s některými jinými jazyky programování kvůli svému dynamickému typování a interpretaci kódu.

Omezená škálovatelnost: Některé frameworky Python mohou mít omezenou škálovatelnost při zpracování velkého počtu požadavků nebo dat.

Omezené možnosti pro vysoké zatížení: Python může mít omezené možnosti zpracování vysokého zatížení a paralelních požadavků ve srovnání s některými jinými jazyky.

Závislost na interpretu: Aplikace v Pythonu vyžadují instalaci interpretu Pythonu na cílovém systému, což může způsobit problémy s kompatibilitou a závislostmi při nasazení.

Nevhodný pro výkonné výpočty: Python nemusí být nejlepší volbou pro úlohy vyžadující intenzivní výpočty nebo práci s velkými objemy dat ve srovnání s některými jinými jazyky, jako jsou C++ nebo Java.

- **Dart**

Výhody:

Jednotnost jazyka: Dart lze použít jak pro frontend, tak pro backend, což přispívá k jednotnosti jazyka a usnadňuje vývoj a podporu aplikací.

Vysoký výkon: Dart zajistí vysoký výkon díky své kompilovatelné povaze, což je zvláště důležité pro backend, kde je efektivita provozu aplikace důležitá.

Asynchronní programování: Dart podporuje asynchronní programování, což usnadňuje zpracování velkého počtu paralelních požadavků a zlepšuje odezvu aplikace.

Rozsáhlé funkce jazyka: Dart poskytuje mnoho užitečných funkcí, jako jsou přísné typy, kolekce, vyšší funkce a další, což usnadňuje vývoj komplexních backendových aplikací.

Podpora frameworků: Existují frameworky v Dartu, jako je například Aqueduct, který poskytuje hotové nástroje pro vývoj serverových aplikací, což zjednodušuje a urychluje proces vývoje.

Nevýhody:

Menší komunita a ekosystém: Dart má menší komunitu vývojářů a méně rozvinutý ekosystém ve srovnání s více běžně používanými jazyky, což může ztížit hledání řešení a podporu.

Omezená podpora externích knihoven: V Dartu může být omezený výběr externích knihoven a nástrojů ve srovnání s jinými jazyky, což může vyžadovat více času a úsilí při vývoji a implementaci určitých funkcí.

Nižší adaptace: Dart není tak široce používaný jako jiné jazyky, jako je JavaScript, Python nebo Java, což může způsobit problémy při hledání kvalifikovaných vývojářů nebo zajištění podpory a školení.

Omezená integrace: Některé nástroje a platformy mohou mít omezenou nebo chybějící podporu pro Dart, což může způsobit problémy při integraci s existujícími systémy nebo infrastrukturou.

Relativní novost: Dart je relativně nový jazyk, což může vyvolávat nejistotu nebo pochybnosti o jeho stabilitě a spolehlivosti ve srovnání s více zavedenými jazyky. (6)

5.2 Praktické srovnání programovacích jazyků

5.2.1 Front-endové programovací jazyky

Prvním kritériem, které bude použito pro srovnání, je výkonnost programovacích jazyků. Je však důležité poznamenat, že v tomto kritériu nebudou zohledněny programovací jazyky HTML a CSS. HTML a CSS nejsou programovací jazyky v pravém slova smyslu, ale značkovací a stylovací jazyky. Jazyk HTML se používá k definování struktury webových stránek a jejich obsahu, zatímco CSS se používá k definování vzhledu těchto stránek. Oba tyto jazyky nemají vlastnosti programovacích jazyků, jako

jsou podmínky, cykly a funkce. Hodnotí se jejich schopnost vytvářet strukturu a vzhled webových stránek, nikoliv jejich výkon. To znamená, že jsou hodnoceny podle toho, jak dobře umožňují navrhovat a prezentovat webové stránky, nikoli podle rychlosti provádění kódu.

K otestování výkonu jazyků JavaScript a TypeScript byl napsán kód, který hledá Fibonacciho čísla pomocí 10 000 iterací. Fibonacciho čísla jsou posloupností, kde každé číslo je součtem dvou předchozích čísel (např. 0, 1, 1, 2, 3, 5 atd.). Tento kód umožňuje vyhodnotit efektivitu provádění algoritmu v různých programovacích jazycích, porovnat rychlost provádění a určit, který z nich je pro danou úlohu vhodnější.

Obr. 1 Kód pro vyhledávání Fibonacciho čísel v jazyce JavaScript

Zdroj: Vlastní zpracování

Obr. 2 Kód pro vyhledávání Fibonacciho čísel v jazyce TypeScript

Zdroj: Vlastní zpracování

Po spuštění kódu byly výsledky následující - JavaScript provedl kód za 2,057 sekundy, TypeScript za 2,019 sekundy.

Aby bylo možné na konci práce určit vedoucí programovací jazyky, bylo rozhodnuto zavést bodový systém pro jednotné hodnocení. Počet bodů tak bude záviset na počtu programovacích jazyků vybraných pro srovnání v daném směru. V této kapitole práce budou posuzovány programovací jazyky JavaScript, TypeScript, HTML a CSS. Jak

však bylo napsáno výše, porovnávat HTML a CSS z hlediska rychlosti provádění kódu nemá smysl.

Po získání výsledků tedy vznikne následující tabulka:

Tab. 1 Počet bodů, kritérium "Výkonnost"

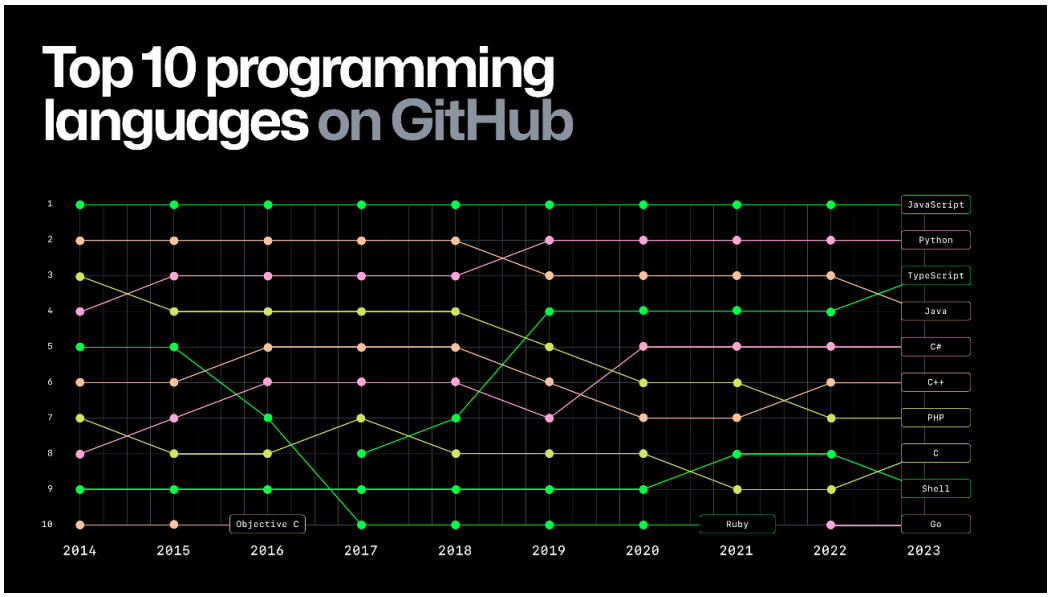
Název programovacího jazyka	Rychlost provádění kódu (sec)	Počet bodů
TypeScript	2.019	4
JavaScript	2.057	3
HTML	-	2
CSS	-	2

Zdroj: Vlastní zpracování

Druhým kritériem, podle kterého budou programovací jazyky porovnávány, je kritérium "Aktivita komunity".

Pro zjištění popularity programovacích jazyků bylo rozhodnuto obrátit se na jeden z nejautoritativnějších zdrojů dat v oblasti vývoje softwaru - web GitHub. Důvodem této volby je široké využívání webu GitHub jako platformy pro ukládání, spolupráci a sdílení kódu mezi vývojáři po celém světě. Analýza statistik na serveru GitHub umožňuje získat objektivní představu o tom, jaké programovací jazyky se aktivně používají v moderním vývoji, a také sledovat dynamiku změn v jejich popularitě.

Obr. 3 Statistika oblíbenosti programovacích jazyků front-end



Zdroj: Github (10)

Z těchto statistických údajů tedy vyplývá, že nejpopulárnějším programovacím jazykem pro rok 2023 bude JavaScript. Na druhém a třetím místě je Python a TypeScript.

Popularita JavaScriptu je dána několika faktory: prvním z nich je, že se jedná o jeden z nejstarších programovacích jazyků pro vývoj webových stránek, což mu dává velké množství zkušeností a stability. Za druhé je JavaScript známý svou jednoduchostí a relativní snadností učení, díky čemuž je přístupný širokému okruhu vývojářů. Kromě toho má JavaScript velkou komunitu vývojářů a obrovské množství knihoven a frameworků, což z něj činí atraktivní volbu pro různé projekty.

Na druhou stranu TypeScript, který je relativně novým jazykem, zatím nedosáhl tak široké popularity jako JavaScript. Pozornost vývojářů však přitahuje díky statickému typování, které zvyšuje robustnost kódu a zlepšuje proces vývoje rozsáhlých projektů. Kvůli své relativní novosti však může mít TypeScript ve srovnání s JavaScriptem méně knihoven a frameworků, což může být pro některé projekty překážkou. Nicméně s růstem jeho popularity lze očekávat rozšiřující se ekosystém a zvýšenou podporu různých knihoven a nástrojů.

HTML a CSS jsou sice základními technologiemi pro vývoj webových stránek, ale obvykle nejsou řazeny mezi front-end programovací jazyky, a to z několika důvodů: první je, že jazyky HTML a CSS jsou z velké části zodpovědné za strukturu a vizuální design webových stránek, nikoli za logiku a funkčnost, což z nich činí méně programovací jazyky ve srovnání s jazykem JavaScript, který je ve velké míře využíván pro interaktivitu a dynamické chování webových stránek. Za druhé, že HTML a CSS jsou

často považovány za doplňkové dovednosti nebo sady nástrojů, které nemusí nutně mít vztah k programování v užším slova smyslu. Spíše jsou považovány za základy tvorby webových stránek, které by měl znát téměř každý vývojář, ale nevyžadují stejnou úroveň algoritmického a logického myšlení jako například programovací jazyky pro všeobecné použití. Obecně platí, že HTML a CSS hrají při vývoji webových stránek klíčovou roli, ale vzhledem k jejich specifičnosti a úloze se o nich obvykle neuvažuje v kontextu hodnocení programovacích jazyků.

Takto vznikne následující tabulka:

Tab. 2 Počet bodů, kritérium "Aktivita komunity"

Název programovacího jazyka	Počet bodů
JavaScript	4
TypeScript	3
HTML	2
CSS	2

Zdroj: Vlastní zpracování

Dalším kritériem, podle kterého budou programovací jazyky hodnoceny, budou " vlastnosti propojení s back-endem".

Kritérium "vlastnosti propojení s back-endem" je subjektivní, protože závisí na konkrétních potřebách a vlastnostech projektu a také na preferencích a zkušenostech vývojáře. Například v jednom projektu může být preferováno použití jazyka JavaScript pro back-end propojení kvůli jeho široké dostupnosti a výkonným nástrojům, zatímco v jiném projektu může být zvolen TypeScript kvůli jeho statickému typování a schopnosti detekovat chyby v době kompilace. Kromě toho může být v některých případech webová aplikace sestavena tak, že se pro interakci s back-endem prostřednictvím vykreslování na straně serveru nebo jiných metod používají jazyky HTML a CSS.

Pro objektivní vyhodnocení tohoto kritéria se nelze obrátit na statistiku, protože statistika nemusí zohlednit specifické souvislosti a vlastnosti každého projektu. Vývoj webu často vyžaduje individuální přístup a to, co může být ideálním řešením pro jeden projekt, může být pro jiný projekt neefektivní nebo nevyhovující. Volba programovacích

jazyků a metod interakce s back-endem tedy závisí na mnoha faktorech, které nemusí být ve statistice vždy zohledněny.

Na prvním místě je JavaScript, který je hlavním programovacím jazykem pro vývoj webových stránek a často se používá k interakci s back-endem prostřednictvím requestů AJAX, rozhraní RESTful API a dalších metod. JavaScript se široce používá k vytváření dynamických a interaktivních webových aplikací prostřednictvím komunikace se serverem a zpracováním přijatých odpovědí.

Po jazyku JavaScript následuje jazyk TypeScript. TypeScript je nadsada jazyka JavaScript, která přidává statické typování a další funkce, které zlepšují vývoj rozsáhlých a složitých projektů. Ačkoli lze jazyk TypeScript používat k psaní kódu front-endu, může být díky svým pokročilým funkcím a podpoře statického typování také efektivním nástrojem pro interakci s back-endem.

Na posledním místě jsou jazyky HTML a CSS. Přestože jsou nedílnou součástí vývoje webu a jsou zodpovědné za strukturu a vizuální vzhled webových stránek, jen zřídkakdy komunikují přímo s back-endem. V podstatě se HTML a CSS používají k zobrazení dat přijatých z back-endu, ale neplní funkci přímé interakce se serverem.

Vznikne tak následující tabulka pořadí:

Tab. 3 Počet bodů, kritérium "Vlastnosti propojení s back-endem"

Název programovacího jazyka	Počet bodů
JavaScript	4
TypeScript	3
HTML	2
CSS	2

Zdroj: Vlastní zpracování

Posledním kritériem, podle kterého budou programovací jazyky porovnávány, je "kompatibilita s prohlížeči".

JavaScript je základním jazykem pro vývoj webových stránek a je vysoce kompatibilní s většinou moderních prohlížečů. Je to proto, že JavaScript je jediným jazykem, který je přímo spouštěn prohlížeči. Většina moderních prohlížečů podporuje JavaScript ve svých nejnovějších verzích a aktivně sleduje jeho standardizaci prostřednictvím mezinárodní organizace ECMA.

TypeScript, přestože je nadstavbou jazyka JavaScript, se před spuštěním zkompiluje do běžného jazyka JavaScript. Obecně je TypeScript kompatibilní se stejnými prohlížeči jako JavaScript, protože konečným výsledkem je obyčejný JavaScript. Při použití nových funkcí jazyka TypeScript nebo typů specifických pro jazyk TypeScript však mohou nastat problémy s kompatibilitou se staršími prohlížeči.

Často se stává, že starší verze prohlížečů nemusí podporovat nové funkce ECMAScriptu v kódu TypeScript. To může způsobit problémy s kompatibilitou, zejména pokud se používají nové metody pole, sliby nebo funkce šipek, které starší prohlížeče neumožňují. Nicméně moderní verze většiny populárních prohlížečů již obvykle podporují tyto nové funkce, a tak problémy s kompatibilitou jsou vzácné.

Internet Explorer (IE), zejména starší verze jako IE 11 a starší, může být příkladem prohlížeče, který nemusí plně podporovat nové metody pole v jazyce TypeScript. Opera Mini, která je často používána na mobilních zařízeních, také může mít svá omezení v podpoře moderních webových standardů a nových funkcí ECMAScriptu, a tím může způsobit problémy s kompatibilitou webových aplikací vytvořených pomocí TypeScriptu.

Prohlížeč Safari na zařízeních iOS, zejména starší verze, také může mít problémy s podporou nových funkcí ECMAScriptu. Stejně tak i UC Browser, který je oblíbený na mobilních zařízeních, nemusí plně podporovat nové funkce v jazyce TypeScript.

HTML je značkovací jazyk, který prohlížeče interpretují a vytvářejí z něj strukturu webových stránek. Ve srovnání s JavaScriptem má jazyk HTML ještě vyšší kompatibilitu s prohlížeči, protože prohlížeče obvykle podporují všechny základní prvky a atributy jazyka HTML, aby byla zajištěna kompatibilita se stávajícími webovými stránkami.

CSS se používá ke stylování a navrhování webových stránek. Je také vysoce kompatibilní s prohlížeči, ale někdy mohou existovat drobné rozdíly v tom, jak různé prohlížeče interpretují a vykreslují určité styly. To může vést k drobným rozdílům v zobrazení mezi různými prohlížeči.

Konečná tabulka je proto následující:

Tab. 4 Počet bodů, kritérium "Kompatibilita s prohlížeči"

Název programovacího jazyka	Počet bodů
HTML	4
JavaScript	3

TypeScript	2
CSS	2

Zdroj: Vlastní zpracování

Pro závěr porovnání programovacích jazyků front-endu lze získat následující tabulku:

Tab. 5 Konečné výsledky srovnání

Název programovacího jazyka	výkon	Aktivita komunity	vlastnosti propojení s back-endem	kompatibilita s prohlížeči	Celkem
JavaScript	3	4	4	3	14
TypeScript	4	3	3	2	12
HTML	2	2	2	4	10
CSS	2	2	2	2	8

Zdroj: Vlastní zpracování

Na základě výsledků srovnání a součtů v tabulce lze konstatovat, že JavaScript je nejvhodnějším programovacím jazykem pro tvorbu front-endové části webových aplikací. Na jedné straně je skutečně pravda, že JS je nejžádanějším programovacím jazykem na trhu tvorby webových aplikací. Na druhou stranu nelze říci, že by TS na svého konkurenta výrazně ztrácel. Pravděpodobně se v krátké době, až se TS mnohem více rozvine, dostane na stejnou úroveň popularity jako JS. Nezapomínejte také, že i když HTML a CSS získaly nejnižší skóre, stále se jedná o oblíbené nástroje pro tvorbu webových aplikací.

5.2.2 Back-endové programovací jazyky

V této kapitole se bude srovnávat back-end programovací jazyky na základě vybraných kritérií. Jedním z kritérií, které bude posuzováno jako první, je výkon.

Pro porovnání programovacích jazyků, jako jsou Python, Go, C#(C-sharp), Java, PHP a Dart, byl napsán stejný algoritmus pro hledání Fibonacciových čísel.

Obr. 4 Kód pro hledání Fibonacciho čísel v Pythonu

practice.py

```python
fib1 = fib2 = 1

n = 10000

for i in range(2, n):
    fib1, fib2 = fib2, fib1 + fib2
    print(fib2, end=' ')

# 1.491 sec
```

Zdroj: Vlastní zpracování

Obr. 5 Kód pro hledání Fibonacciho čísel v Go

practice.go

```go
package main

import "fmt"

func main() {
    var n2, n1, n = 0, 1, 10000
    for i := 2; i <= n; i++ {
        n2, n1 = n1, n1+n2
        fmt.Println(n2)
    }

}

// 0.21 sec
```

Zdroj: Vlastní zpracování

Obr. 6 Kód pro hledání Fibonacciho čísel v C-Sharp

practice.cs

```
using System;

namespace Practice
{
    public class Program
    {
        public static void Main(string[] args)
        {
            int a = 1, b = 1, c = 0, len = 10000;
            for (int i = 1; i < len; i++)
            {
                c= a + b;
                Console.Write(" {0}", c);
                a= b;
                b= c;
            }
        }
    }
}

// 0.25 sec
```

Zdroj: Vlastní zpracování

Obr. 7 Kód pro hledání Fibonacciho čísel v Javě

practice.java

```
import java.util.*;
import java.lang.*;

class Practice
{
    public static void main(String args[])
    {
     int n1=1,n2=1,n3,i,count=10000;

     for(i=1;i<count;++i)
        {
         n3=n1+n2;
         System.out.print(" "+n3);
         n1=n2;
         n2=n3;
        }
    }
}

// 0.33 sec
```

Zdroj: Vlastní zpracování

Obr. 8 Kód pro hledání Fibonacciho čísel v PHP

practice.php

```php
<?php

$x = 1;
$y = 1;
$n = 10000;

for($i=0;$i<=$n;$i++)
{
    $z = $x + $y;
    echo $z."\n";
    $x=$y;
    $y=$z;
}
    // 0.22 sec
?>
```

Zdroj: Vlastní zpracování

Obr. 9 Kód pro hledání Fibonacciho čísel v Dartu

practice.dart

```dart
void main() {
  int a=1;
  int b=1;
  int n=10000;
  print(a);
  print(b);
  for(int i=1;i<n; i++){
    int c=a+b;
    print(c);
    a=b;
    b=c;
  }
}

// 1.17 sec
```

Zdroj: Vlastní zpracování

Kód byl záměrně napsán v přibližně stejné velikosti a počtu řádků, aby se vyloučila možnost, že velikost kódu ovlivní rychlost provádění.

Po spuštění výsledků lze zjistit, že nejrychlejším prezentovaným programovacím jazykem je Go, nejpomalejším Python. Celkově byla získána následující tabulka:

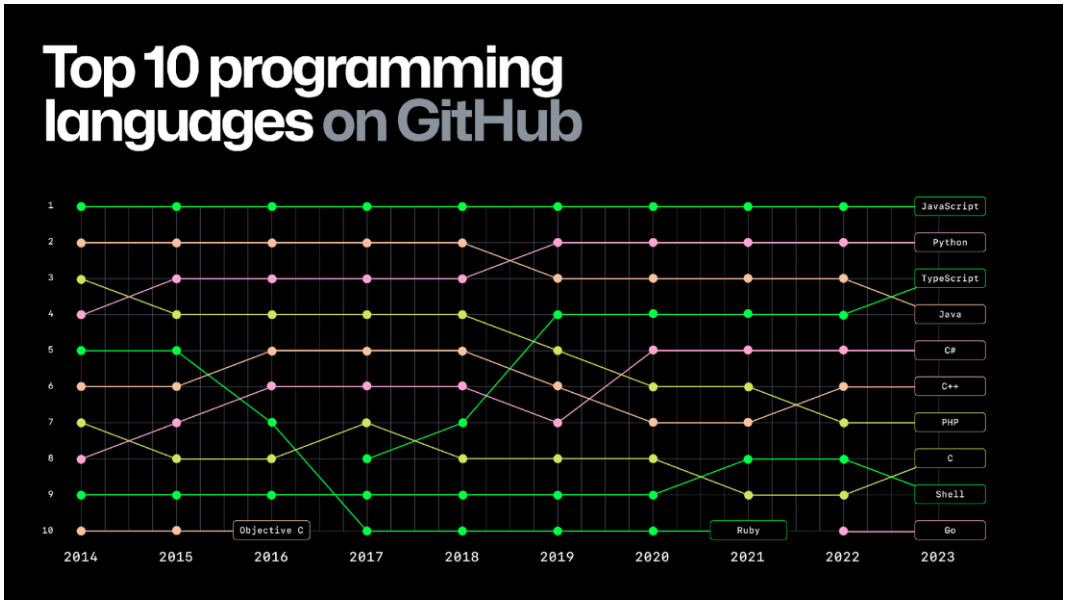
Tab. 6 Počet bodů, kritérium "Výkonnost"

Název programovacího jazyka	Rychlost provádění kódu (sec)	Počet bodů
Go	0.21	6
PHP	0.22	5
C#	0.25	4
Java	0.33	3
Dart	1.17	2
Python	1.491	1

Zdroj: Vlastní zpracování

Dalším kritériem, které bude použito pro porovnání programovacích jazyků, je aktivita a podpora komunity. V této práci bude aktivita komunity analyzována na základě údajů z GitHubu, jednoho z nejpopulárnějších zdrojů mezi programátory.

Obr. 10 Statistika oblíbenosti programovacích jazyků front-end



Zdroj: Github (10)

Statistika pro rok 2023 ukazuje, že Python je na prvním místě v seznamu nejoblíbenějších programovacích jazyků na GitHubu díky širokému spektru aplikací, snadnému učení a masivnímu využití v různých oblastech programování. Java, která je na druhém místě, si udržuje svou popularitu díky obrovskému počtu existujících projektů a firemní podpoře. Jazyk C# zaujímá třetí místo díky aktivní podpoře společnosti Microsoft a jeho využití při vývoji aplikací pro systém Windows. Jazyk PHP, který se umístil na čtvrtém místě, zůstává mezi webovými vývojáři oblíbený i přes rostoucí konkurenci. Jazyky Go a Dart obsadily poslední místa. Je to dáno jejich novostí, relativní neznámostí a nepoužíváním v době analýzy.

Z výsledků analýzy vyplývá následující tabulka:

Tab. 7 Počet bodů, kritérium "Aktivita a podpora komunity"

Název programovacího jazyka	Počet bodů
Python	6
Java	5
C#	4
PHP	3
Go	2
Dart	1

Zdroj: Vlastní zpracování

Dalším kritériem, které bude použito pro porovnání programovacích jazyků, je jejich škálovatelnost. Toto kritérium hraje důležitou roli při výběru programovacího jazyka pro vývoj rozsáhlých a složitých projektů. Škálovatelnost definuje schopnost jazyka vyrovnat se s nárůstem velikosti kódu, počtu uživatelů a požadavků na systém, a přitom nezhoršit výkon a celkovou stabilitu. Při porovnávání programovacích jazyků podle tohoto kritéria budou brány v úvahu jejich schopnosti v oblasti paralelismu, podpory distribuovaných výpočtů, efektivního využívání zdrojů a snadného škálování ve vertikálním i horizontálním směru. Dostupnost dobré škálovatelnosti programovacího jazyka může významně ovlivnit dobu vývoje, celkový výkon a efektivitu nasazení aplikace, proto je toto kritérium důležité pro porovnání a výběr programovacího jazyka v závislosti na potřebách projektu.

Java je známá svou vysokou škálovatelností díky virtuálnímu stroji Java Virtual Machine (JVM) a principu "napiš jednou, spusť kdekoli". To umožňuje vyvíjet aplikace, které mohou běžet na různých platformách s minimálními změnami.

Python je také vysoce škálovatelný díky své flexibilitě a rozsáhlému ekosystému knihoven. Je široce používán pro vývoj různých aplikací včetně webových služeb, vědeckých výpočtů, analýzy dat a mnoha dalších.

Jazyk C# nabízí dobrou škálovatelnost pro vývoj podnikových aplikací a herních projektů, zejména při použití platformy .NET. Podporuje vývoj rozsáhlých a komplexních aplikací s využitím objektově orientovaného přístupu.

PHP: Jazyk PHP je dobře škálovatelný pro vývoj webových aplikací a webových stránek, zejména díky rozšířenému používání webových frameworků a CMS, jako jsou WordPress, Drupal a Joomla. Je třeba také poznamenat, že tento programovací jazyk se většinou používá k psaní monolitů, což je architektonický přístup, při kterém se celý softwarový produkt nebo aplikace vyvíjí a nasazuje jako celek, aniž by byl rozdělen na samostatné moduly nebo komponenty, a modelů MVC - Model-View-Controller, což je populární návrhový vzor používaný při vývoji softwaru k organizaci kódu a zjednodušení procesu vývoje.

Jazyk Go je známý svou efektivitou a jednoduchostí, díky čemuž je vhodný pro vytváření distribuovaných systémů a mikroslužeb. Jeho škálovatelnost však může být ve srovnání s ostatními jazyky v tomto seznamu poněkud omezená. Jednou z klíčových vlastností jazyka Go je podpora asynchronního programování, díky čemuž je ideální pro zpracování velkého počtu souběžných požadavků. Konkurenční a paralelní mechanismy jazyka Go umožňují efektivní využití vícejádrových procesorů a serverových zdrojů, což zajišťuje vysoký výkon aplikací.

V době analýzy může být škálovatelnost Dartu méně výrazná kvůli jeho relativní novosti a omezenému použití v různých vývojových doménách.

Mezi nejlépe škálovatelné programovací jazyky v tomto seznamu lze tedy zařadit jazyky Java, Python a C#, zatímco jazyky Go a PHP mají také dobrou škálovatelnost, ale v jiných kontextech. Dart, jakožto nový jazyk, nemusí zatím dosahovat stejné úrovně škálovatelnosti jako jeho konkurenti.

Tab. 8 Počet bodů, kritérium "Škálovatelnost"

Název programovacího jazyka	Počet bodů
Java	6
Python	5
C#	4
PHP	3
Go	2
Dart	1

Zdroj: Vlastní zpracování

Seřadit programovací jazyky od nejlepšího po nejhorší v kontextu bezpečnosti může být obtížný úkol, protože každý jazyk má v tomto ohledu své silné a slabé stránky. Nicméně s ohledem na řadu faktorů, včetně přítomnosti vestavěných bezpečnostních mechanismů, rozšířenosti v kritických oblastech a úrovně historických zranitelností, lze navrhnout následující top:

Jazyk Go je často považován za jeden z nejbezpečnějších programovacích jazyků díky svým vestavěným bezpečnostním mechanismům, jako jsou:

- **Přísné typování** - Go nabízí striktní statické typování, které umožňuje odhalit chyby v době kompilace, včetně chyb souvisejících s bezpečností;
- **Kontrola hranic pole** - Go nemá automatický přístup do paměti, což snižuje pravděpodobnost výskytu chyb, jako je přetečení bufferu;
- **Segmentace paměti**: Go neposkytuje přímý přístup k ukazatelům ani explicitní manipulaci s pamětí

Java také poskytuje mnoho bezpečnostních nástrojů a její použití v kritických systémech, jako jsou bankovní aplikace, potvrzuje její vysokou úroveň zabezpečení. Mezi vestavěné bezpečnostní nástroje patří:

- **Správa paměti**: Java nemá žádnou ruční správu paměti, což pomáhá předcházet únikům paměti a dalším chybám při správě zdrojů.
- **Kontrola typů**: Java poskytuje přísnou kontrolu typů při kompilaci, což pomáhá předcházet mnoha chybám, včetně bezpečnostních.
- **Zabezpečené třídy**: Některé třídy jazyka Java, například třídy pro práci s vlákny a vícevláknové třídy, poskytují bezpečnostní mechanismy, které

zabraňují závodům dat a dalším problémům spojeným s paralelním prováděním.

Jazyk C# má podobné bezpečnostní mechanismy jako Java:

- **Striktní typování**: Stejně jako Java nabízí i C# striktní statické typování, které umožňuje odhalit chyby již při kompilaci.
- **Správa paměti**: Jazyk C# je rovněž založen na principu rubbish collection, který snižuje riziko úniku paměti a dalších problémů se správou paměti.
- **Zabezpečené třídy a knihovny**: Platforma .NET poskytuje širokou škálu zabezpečených tříd a knihoven pro práci se soubory, síťování, šifrování a další úlohy, které vývojářům pomáhají vytvářet bezpečné aplikace.

Python lze rovněž považovat za relativně bezpečný programovací jazyk, zejména při správném použití a zpracování uživatelských vstupů. Má následující bezpečnostní nástroje:

- **Dynamické typování s typovou kontrolou**: Python používá dynamické typování s možností kontroly typů, které pomáhá odhalovat chyby za běhu.
- **Vestavěné bezpečnostní knihovny**: Python poskytuje širokou škálu vestavěných knihoven pro zpracování dat, síťování a šifrování, které pomáhají snižovat riziko zranitelností.

V moderních verzích a se správným přístupem může PHP poskytnout přijatelnou úroveň zabezpečení, i když má určité bezpečnostní problémy způsobené historickými zranitelnostmi.

Vzhledem k tomu, že Dart je relativně nový jazyk a má omezené zkušenosti s kritickými aplikacemi, může být jeho pozice na tomto seznamu méně jistá. Vzhledem k některým bezpečnostním mechanismům, jako je striktní typování, lze však Dart považovat i za relativně bezpečný jazyk.

Takto vznikne následující tabulka pořadí:

Tab. 9 Počet bodů, kritérium "Bezpečnost"

Název programovacího jazyka	Počet bodů
Go	6
Java	5
C#	4
Python	3

PHP	2
Dart	1

Zdroj: Vlastní zpracování

Po porovnání programovacích jazyků podle všech uvedených kritérií vznikla následující souhrnná tabulka:

Tab. 10 Konečný výsledek porovnání

Název programovacího jazyka	výkon	aktivita a podpora komunity	škálovatelnost	bezpečnost	Celkem
Java	3	5	6	5	19
Go	6	2	2	6	16
C#	4	4	4	4	16
Python	1	6	5	3	15
PHP	5	3	3	2	13
Dart	2	1	1	1	5

Zdroj: Vlastní zpracování

Závěrem této kapitoly je, že každý ze zkoumaných programovacích jazyků má jedinečné vlastnosti a výhody, které mohou být klíčové v konkrétních scénářích použití. Java, která se v tomto srovnání umístila na prvním místě, se ukázala jako vynikající volba díky své vysoké škálovatelnosti, vynikajícímu zabezpečení a aktivní komunitě vývojářů. Na druhém místě se umístil jazyk Go, který je jedním z nejbezpečnějších a nejproduktivnějších programovacích jazyků. Jazyk C#, který se umístil na třetím místě, je vyváženým jazykem, který kombinuje mnoho pozitivních aspektů. Na čtvrtém místě je Python, který je velmi populární, ale ve srovnání s ostatními je to relativně pomalý jazyk. Jazyk PHP na pátém místě se potýká s problémy zastaralosti a omezené škálovatelnosti. A konečně poslední místo v žebříčku obsadil Dart, který i přes některé své přednosti nemohl v souboru kritérií konkurovat ostatním jazykům.

6 Závěr

V závěru bakalářské práce byla provedena srovnávací studie programovacích jazyků pro psaní webových stránek. V průběhu práce byly programovací jazyky rozděleny do dvou skupin - front-end a back-end a byla zvažována hlavní kritéria hodnocení jazyků, jako je výkon, aktivita komunity, škálovatelnost, bezpečnost, funkce propojení back-endu a kompatibilita s prohlížeči. Každý z jazyků byl analyzován na základě těchto kritérií, což vedlo k určení jejich silných a slabých stránek.

Výsledkem tohoto výzkumu bylo zjištění, že nejvhodnějšími programovými jazyky pro front-end jsou TypeScript a JavaScript. Tyto programovací jazyky jsou skutečně populárnější než ostatní, a jsou více používané při psaní front-endů, což dokládají statistiky serveru GitHub. Na posledních místech se umístily HTML a CSS, nicméně nelze říci, že by se tyto jazyky pro vývoj front-endu nepoužívaly. Jejich profil je však poměrně úzký a obvykle se tyto jazyky používají ve spojení s JS nebo TS.

Nejvhodnějšími programovými jazyky pro back-end jsou Java, Go a C#. Java právem zaujímá první místo a skutečně dnes patří mezi nejoblíbenější jazyky při výběru jazyka pro psaní back-endu. Go neboli Go-lang je nový jazyk vytvořený společností Google, který si však rychle získává své publikum díky tomu, že byl vytvořen speciálně pro psaní back-endů, a proto má dobrý výkon a bezpečnost, což jsou nejoblíbenější požadavky na jazyky pro back-endy. Jazyk C# se nachází uprostřed žebříčku a je skutečně vyváženým jazykem vhodným pro různé úlohy. Na posledních místech se umístily jazyky Python, PHP a Dart. Python, ačkoli je dnes populární, má některé dost podstatné nuance - nízký výkon a poměrně slabé zabezpečení. PHP je stárnoucí jazyk, ke kterému se vývojáři obracejí stále méně, když mají po ruce silnější nástroje, jako je Go. Dart zaujímá poslední místo díky své novosti, takže je možné, že v blízké budoucnosti zaujme své místo mezi nejoblíbenějšími programovacími jazyky a stane se jedním z nejsilnějších nástrojů pro vývoj.

Na závěr je třeba poznamenat, že ačkoli byla sestavena tabulka hodnocení programovacích jazyků pro front-end a back-end, nelze jednoznačně říci, že jeden jazyk je lepší nebo horší než druhý. Je třeba vzít v úvahu, že každý programovací jazyk má svou vlastní sadu nástrojů, která jediná může být vhodná pro určité úkoly, které je třeba provést. Proto je výběr nejvhodnějšího programovacího jazyka ponechán na vývojáři aplikace.

7 Seznam použitých zdrojů

1) Nicholas C. *Zakas Professional JavaScript for Web Developers*. Wiley, 2005. ISBN: 9780764579080
2) Josh Goldberg. *Learning TypeScript*. O'Reilly Media, 2022. ISBN: 9781098110284 [cit. 2023 – 11 – 23]
3) Vertex Academy. *Historie vývoje jazyka HTML*. [online] 17.06.2016 [cit. 2023 – 11 – 23] Dostupné z: https://vertex-academy.com/tutorials/ru/html_history/
4) Mike McGrath. *GO Programming in easy step.* In Easy Steps Limited, 2020. ISBN: 9781840789270
5) Peter Sestoft, Henrik I. Hansen. *C# Precisely*. MIT Press, 2012. ISBN: 9780262516860.
6) Moises Belchin, Patricia Juberias. *Web Programming with Dart*. Apress, 2015. ISBN: 9781484205563
7) Adam Hopkins. *Python Web Development with Sanic*. Packt Publishing, 2022. ISBN: 9781801816434
8) Budi Kurniawan. *Java.* Brainy Software, 2015. ISBN: 9780992133047
9) Mark Lassoff. *Java Programming for Beginners.* Packt Publishing, 2017. ISBN: 9781788299046
10) Github. *Octoverse: The state of open source and rise of AI in 2023.* [online] 2023 [cit. 2023 – 11 – 23] Dostupné z: Octoverse: The state of open source and rise of AI in 2023 - The GitHub Blog
11) Julie C. Meloni, Jennifer Kyrnin. *HTML, CSS, and JavaScript All in One*. Pearson Education, 2018. ISBN: 9780135167076
12) Typescript Publishing. TypeScript Programming Language. Independently Published, 2019. ISBN: 9781708839802
13) Kirupa Chinnathambi. JavaScript Absolute Beginner's Guide. Pearson Education, 2019. ISBN: 9780136204350
14) Harry. H. Chaudhary. C# Programming. CreateSpace Independent Publishing Platform, 2014. ISBN: 9781500193461

8 Seznam obrázků

9 Seznam tabulek