**Česká zemědělská univerzita v Praze**

**Provozně ekonomická fakulta**

**Katedra informačních technologií**

# Diplomová práce

## Využití Front-End frameworků při tvorbě webové stránky

**Bc. Vojtěch Balata**

# ČESKÁ ZEMĚDĚLSKÁ UNIVERZITA V PRAZE

Provozně ekonomická fakulta

# ZADÁNÍ DIPLOMOVÉ PRÁCE

Bc. Vojtěch Balata

Informatika

Název práce

**Využití Front-End frameworků při tvorbě webové stránky**

Název anglicky

**Use of Front-End Frameworks for creating Websites**

---

**Cíle práce**

Hlavním cílem diplomové práce bude realizace vhodného porovnání jednotlivých frameworků sloužících k vytváření webových stránek z hlediska vývoje UI.
V teoretické části budou popsány metodiky a postupy CSS a JavaScript frameworků, které lze uplatnit při tvorbě webových stránek. V oblasti vývoje front-end prostředí bude kladen důraz na popis nástrojů, kterými zvolené frameworky disponují.
Dílčím cílem praktické části diplomové práce bude, na základě syntézy zjištěných poznatků, vytvořit webové stránky pomocí různých metod a frameworků, které je možné při vývoji použít. Následně budou použité frameworky porovnány na základě vhodně stanovených kritérií.

**Metodika**

Metodika diplomové práce bude založena na studiu a analýze odborných informačních zdrojů zabývajících se zvolenou problematikou. Praktická část práce bude zaměřena na vývoj funkční webové stránky pomocí zvolených front-end frameworků. Vytvořené verze webové stránky budou otestovány a porovnány na základě vhodně stanovených kritérií. Syntézou poznatků teoretické a praktické části práce budou formulovány závěry diplomové práce.

**Doporučený rozsah práce**

60 – 80 stran

**Klíčová slova**

JavaScript, CSS, HTML, framework, front-end, web, UI

---

**Doporučené zdroje informací**

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

HOGAN, Brian P., 2011. HTML5 a CSS3: výukový kurz webového vývojáře. Brno: Computer Press. ISBN 978-80-251-3576-1.

KADLEC, Tim, 2014. Responzivní design profesionálně. Brno: Zoner Press. Encyklopedie Zoner Press. ISBN 978-80-7413-280-3.

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

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

---

**Předběžný termín obhajoby**

2022/23 LS – PEF

**Vedoucí práce**

Ing. Petr Benda, Ph.D.

**Garantující pracoviště**

Katedra informačních technologií

Elektronicky schváleno dne 14. 7. 2022

**doc. Ing. Jiří Vaněk, Ph.D.**

Vedoucí katedry

Elektronicky schváleno dne 28. 11. 2022

**doc. Ing. Tomáš Šubrt, Ph.D.**

Děkan

V Praze dne 30. 03. 2024

**Čestné prohlášení**

Prohlašuji, že svou diplomovou práci "Využití Front-End frameworků při tvorbě webové stránky" jsem vypracoval samostatně pod vedením vedoucího diplomové 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 diplomové práce dále prohlašuji, že jsem v souvislosti s jejím vytvořením neporušil autorská práva třetích osob.

V Praze dne 30. 3. 2024

**Poděkování**

Rád bych touto cestou poděkoval panu Ing. Petr Bendovi, Ph.D. za vedení mé diplomové práce a odbornou pomoc, cenné rady a za jeho čas, který mi věnoval pro potřeby zpracování diplomové práce.

# Využití Front-End frameworků při tvorbě webové stránky


## Abstrakt

Diplomová práce je zaměřena na Front-end design webových stránek, kde se autor primárně zaměřuje na popsání a analýzu běžně používaných JavaScript a UI frameworků. Poznatky z analýzy jsou poté uplatněné v ukázkové webové stránce. Práce je rozdělena na dvě části.
V první části práce jsou popsány teoretické principy problematiky vývoje webových stránek a aplikací, přičemž je u UI frameworků kladen velký důraz na responzivní design, který je v dnešní době používání různě velkých zařízeních stěžejní a tyto nástroje se jím ve větší míře zabývají. Následně jsou popsány postupy týkající se architektury moderních JavaScript frameworků. Na závěr této části je popsána vícekriteriální analýza variant, která je využita při analýze frameworků v druhé části práce.
Druhá část práce je věnována vlastní práci. V této části jsou vytvořeny verze webové stránky pomocí popisovaných nejpoužívanějších frameworků. Pro celkem šest frameworků (tři webové aplikační frameworky, tři UI frameworky) jsou vytvořeny tři webové stránky, které slouží k určení a hodnocení vhodných kritérií. Následně je provedena vícekriteriální analýza variant s cílem nalézt jak u JavaScript, tak i u UI frameworků kompromisní variantu. Vícekriteriální analýza je provedena pomocí metody váženého součtu. Výsledkem této analýzy je, že jako JavaScript framework je vhodné použít Vue a jako UI framework pak Tailwind. Výsledky jsou formulovány a porovnány se závěry z jiných odborných zdrojů.



**Klíčová slova:** html, css, javascript, webové stránky, framework, front-end, web

# Use of Front-End Frameworks for creating Websites

**Abstract**

The thesis focuses on Front-end design of websites, where the author primarily describes and analyzes commonly used JavaScript and UI frameworks. The findings from the analysis are then applied to a sample website. The work is divided into two parts.

In the first part of the work, the theoretical principles of the issue of web development and applications are described, with a great emphasis on responsive design in UI frameworks, which is crucial in today's use of variously sized devices and these tools deal with it to a greater extent. Subsequently, procedures related to the architecture of modern JavaScript frameworks are described. At the end of this part, a multi-criteria analysis of variants is described, which is used in the analysis of frameworks in the second part of the work.

The second part of the work is devoted to the author's own work. In this part, versions of the website are created using the described most commonly used frameworks. For a total of six frameworks (three web application frameworks, three UI frameworks), three websites are created, which serve to determine and evaluate suitable criteria. Subsequently, a multi-criteria analysis of variants is performed with the aim of finding a compromise variant in both JavaScript and UI frameworks. The multi-criteria analysis is performed using the method of the weighted sum. The result of this analysis is that as a JavaScript framework it is appropriate to use Vue and as a UI framework then Tailwind. The results are formulated and compared with conclusions from other professional sources.

**Keywords**: html, css, javascript, websites, framework, front-end, web

# Obsah

# 1 Úvod

V dnešní době, kdy digitalizace a internetové technologie procházejí nebývalým rozmachem, se vývoj webových stránek a aplikací stává stále komplexnějším úkolem. V centru tohoto vývoje stojí Front-End frameworky, které umožňují rychlejší a efektivnější tvorbu uživatelských rozhraní. Tato diplomová práce se zaměřuje na využití těchto frameworků při tvorbě webových stránek, kde klade důraz na analýzu a porovnání běžně používaných JavaScript a UI frameworků. Cílem je nejen poskytnout teoretický základ vývoje webových stránek, ale také prakticky aplikovat získané poznatky při tvorbě ukázkové webové stránky.

V první části práce jsou představeny základní principy a metodiky, které definují současný přístup k vývoji front-endu, s důrazem na responzivní design. Responzivní design je nezbytný pro optimalizaci webových stránek pro různá zařízení, což v dnešní době mobilních zařízení a tabletů představuje klíčový aspekt vývoje. Dále jsou analyzovány architektonické aspekty moderních JavaScript frameworků, které představují základ pro dynamické a interaktivní webové stránky.

Druhá část práce se věnuje praktické aplikaci těchto frameworků. Na základě vícekriteriální analýzy variant je cílem identifikovat nejvhodnější frameworky pro různé typy webových stránek. Tato analýza umožňuje objektivní hodnocení a výběr frameworků na základě stanovených kritérií, což vede k nalezení optimálních řešení pro specifické požadavky projektu. Výsledkem je nejen praktická ukázka využití těchto nástrojů, ale také srovnání a hodnocení jejich výhod a nevýhod v kontextu reálných projektů.

Tato diplomová práce tak přináší komplexní pohled na současné možnosti vývoje webových stránek pomocí Front-End frameworků, od teoretických východisek přes analýzu až po praktické aplikace. S narůstající důležitostí internetových technologií a digitalizace společnosti představuje tento výzkum nejen akademický přínos, ale také praktický návod pro vývojáře a designéry, jak efektivně využívat moderní technologie pro tvorbu responzivních a uživatelsky přívětivých webových stránek.

# 2 Cíl práce a metodika

## 2.1 Cíl práce

Hlavním cílem diplomové práce bude realizace vhodného porovnání jednotlivých frameworků sloužících k vytváření webových stránek z hlediska vývoje UI.
V teoretické části budou popsány metodiky a postupy CSS a JavaScript frameworků, které lze uplatnit při tvorbě webových stránek. V oblasti vývoje front-end prostředí bude kladen důraz na popis nástrojů, kterými zvolené frameworky disponují.
Dílčím cílem praktické části diplomové práce bude, na základě syntézy zjištěných poznatků, vytvořit webové stránky pomocí různých metod a frameworků, které je možné při vývoji použít. Následně budou použité frameworky porovnány na základě vhodně stanovených kritérií.

## 2.2 Metodika

Metodika diplomové práce bude založena na studiu a analýze odborných informačních zdrojů zabývajících se zvolenou problematikou. Praktická část práce bude zaměřena na vývoj funkční webové stránky pomocí zvolených front-end frameworků. Vytvořené verze webové stránky budou otestovány a porovnány na základě vhodně stanovených kritérií. Syntézou poznatků teoretické a praktické části práce budou formulovány závěry diplomové práce.

# 3 Teoretická východiska

## 3.1 Základní technologie pro vývoj webových aplikací

### 3.1.1 Značkovací jazyk HTML

HTML je zkratka pro Hypertext Markup Language, což je jazyk používaný k vytváření webových stránek. Hypertext představuje text s odkazy, které umožňují přecházet mezi různými hypertextovými dokumenty. Tento typ dokumentu může obsahovat různé prvky, jako jsou seznamy, tabulky, obrázky a formuláře. HTML patří do skupiny značkovacích jazyků, které slouží k anotaci dokumentů. Výraz „mark up“ vychází z tradice vydavatelství, kde je zvykem, že redaktoři přidávají k rukopisům poznámky a opravy. Nejnovější verzi HTML označujeme jako HTML5 a píše se toto označení spojeně, nikoli s mezerou. Na obrázku 1 lze najít přehled všech verzí HTML a data jejich vydání (Coremans, 2015, s. 9).

| Year | Version |
|---|---|
| 1989 | Tim Berners-Lee proposed an Internet-based hypertext system. The Web was born. |
| 1991 | Tim Berners-Lee invented HTML. This early version can be considered HTML version 1 and consists of only 18 elements. |
| 1993 | Dave Raggett, a computer scientist, proposed HTML+. |
| 1995 | Tim Berners-Lee published HTML 2.0 as a standard. |
| January 1997 | The publication of HTML 3.2 as a W3C recommendation. |
| December 1997 | The publication of HTML 4.0 as a W3C recommendation. |
| 1999 | The publication of HTML 4.01 as a W3C recommendation. |
| 2000 | The publication of XHTML 1.0 as a W3C recommendation. |
| 2001 | The publication of XHTML 1.1 as a W3C recommendation. |
| 2014 | HTML5 became a W3C standard. |

*Obrázek 1 - Historie HTML (Coremans, 2015, s.10)*

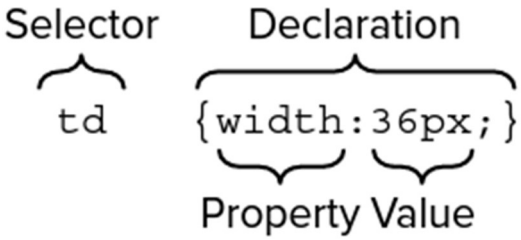
V textových procesorech je možné použít různé formátování textu k vyjasnění struktury dokumentu. Nadpisy můžete odlišit od textu aplikací specifického stylu nadpisu, často ve zvětšené velikosti písma. Pro zahájení nového odstavce slouží klávesa Enter. Do textu lze také vkládat tabulky pro organizaci informací nebo vytvořit seznamy s odrážkami, které shrnují související informace. I když tyto prvky mění vzhled dokumentu, jejich primárním cílem je zajistit jeho přehlednou strukturu a usnadnit tak pochopení obsahu (Larsen, 2013, s. 43).

Při přípravě dokumentů pro internet používáte podobnou techniku jako v textovém editoru, ale místo stylů přidáváte do textu tzv. tagy. U HTML je klíčové, že tagy slouží k určení

struktury dokumentu, nikoliv k jeho vizuální prezentaci. Tagy určují, co je nadpisem, odstavcem, co se zařadí do tabulky a tak dále. Webové prohlížeče jako Internet Explorer, Firefox a Google Chrome pak tuto strukturu používají k tomu, aby text zobrazily uživateli v podobě, na kterou jsou zvyklí z textových procesorů, s většími nadpisy, mezerami mezi odstavci a odrážkami v podobě kroužků před položkami seznamu (Larsen, 2013, s. 43)

### 3.1.2 Kaskádové styly CSS

CSS umožňuje definovat pravidla pro prvky na webové stránce, která řídí způsob, jakým se mají tyto prvky zobrazovat. Například na obrázku 2 je znázorněno pravidlo CSS, které se dělí na dvě hlavní části: selektor určuje, ke kterým prvkům se pravidlo vztahuje (pokud se týká více prvků, je možné vytvořit seznam oddělený čárkami), a deklarace specifikuje, jaký styl se má na tyto prvky aplikovat (Larsen, 2013, s. 232).




*Obrázek 2 - Zápis CSS (Larsen, 2013, s. 232)*

Pravidlo zobrazené na obrázku 2 se týká všech prvků <td> a stanovuje, že jejich šířka by měla být 36 pixelů. Toto nastavení je rozděleno na dvě části oddělené dvojtečkou: první část, "vlastnost", určuje, který atribut vybraných prvků se má změnit, což je v tomto případě šířka. Druhá část, "hodnota", specifikuje, jak by měla být tato vlastnost nastavena, tedy že buňky v tabulce mají být široké 36 pixelů (Larsen, 2013, s. 232).

V dnešní době se klasické kaskádové styly téměř nepoužívají. Čisté CSS totiž v podstatě neumožňuje definovat proměnné, vnořování selektorů nebo například funkce. Tyto nedostatky řeší CSS preprocesory, které kód napsaný právě v nich konvertuje do standartního CSS tak, aby ho mohl zpracovat webový prohlížeč. V době psaní této práce je stabilní verzí CSS3, která má oproti předchůdcům jako CSS a CSS2 značné výhody.

Terra (2023) uvádí, že CSS3 rozšiřuje možnosti designérů o složitější animační vlastnosti jako transformace, přechody a efekty. Vlastnost průhlednosti umožňuje designérům nastavit elementy stránky tak, aby byly částečně nebo úplně průhledné. Můžete nastavit úroveň průhlednosti prvků, aby byly úplně krycí, průhledné nebo dokonce průsvitné. Zaoblené rohy poskytují elementům na stránce sofistikovanější a esteticky příjemnější vzhled. Dříve, před

představením vlastnosti border-radius, museli vývojáři psát dlouhý kód k zaoblení rohů. CSS3 také umožňuje jednoduchou aplikaci textových a boxových stínů, což designérům umožňuje snadno přidávat stíny k textu a definovat jejich barvu, úhel a míru rozostření.
Terra (2026) zároveň popisuje rozdíly mezi původní verzí CSS a CSS3. Původní CSS vyšly roku 1996, zatímco nová verze roku 2005. CSS3 podporuje Mediální dotazy (Media Queries), zatímco původní verze mediální dotazy nepodporuje. Mediální dotazy budou podrobněji popsány v kapitole 3.3.3. CSS3 jsou podporovány všemi moderními webovými prohlížeči, což se nedá tvrdit o původní verzi, která nepodporuje moderní prohlížeče, ale stále podporuje starší verze Exploreru a prohlížeče Google Chrome. Původní verze umožňuje pouze základní typy animací, ale již nedisponuje textovými animacemi, vlastností transformace, tranzicí, nebo 3D animací. Naopak CSS3 těmito všemi zmíněnými vlastnostmi disponuje. Verze z roku 1996 podporuje pouze starý formát barev, zatímco CSS3 nabízí různé gradientní barvy a schémata jako RGBA, HSLA, HSL a podobně. Posledním zmíněným rozdílem je výkon, který je ve stávající verzi v porovnaní se starší verzí vyšší a nevyžaduje tolik paměti.

### 3.1.3 CSS preprocesory

CSS preprocessory se čím dál tím více etablují jako klíčový nástroj v pracovním postupu vývojářů front-endového webu. Vzhledem k tomu, že CSS je jazyk s mnoha složitostmi a jemnostmi, vývojáři se v zájmu zjednodušení práce s ním často obrací k použití preprocessory typu SASS nebo LESS. CSS preprocessory přetvářejí kód napsaný ve speciálním syntaxi za pomocí dedikovaného kompilátoru. Tímto způsobem vzniká CSS soubor, který lze následně vložit do hlavního HTML dokumentu (CSS Preprocessors Explained, 2020).
Proměnné, které jsou běžně užívány v mnoha programovacích jazycích, jsou prvkem, který CSS standardně nenabízí. Proměnné umožňují stanovit hodnotu pouze jednou a následně ji využívat v rámci celého programu nebo stylu (CSS Preprocessors Explained, 2020). Definování proměnné by v SCSS mohlo vypadat následovně:
Smyčky (loop), jež se často objevují v různých programovacích jazycích, jsou dalším prvkem chybějícím v CSS. Umožňují opakovat určitý kód opakovaně, aniž by bylo potřeba jej pokaždé znovu zapisovat (CSS Preprocessors Explained, 2020).

#### 3.1.3.1 SASS

SASS, což je zkratka pro Syntakticky úžasné styly (Syntactically Awesome Style Sheets), je oblíbený nástroj pro rozšíření možností CSS. Kód v SASS se nejdříve zpracuje a poté se zkompiluje do běžného CSS, které slouží k vizuálnímu formátování prvků na webových stránkách. SASS je zodpovědný za to, jak se tyto prvky nakonec zobrazí na stránce. Kód v SASS je podstatně lépe strukturovaný než běžný CSS kód, což umožňuje dosáhnout požadované funkcionality s použitím méně kódu. To je zásadní zejména u velkých projektů, na kterých spolupracuje několik vývojářů. Díky přehlednému stylování je pro každého vývojáře snadné pochopit kód, který napsali a použili jeho kolegové. SASS zavádí do světa CSS koncept znovupoužitelnosti známý z programovacích jazyků. Díky SASS mohou vývojáři využívat proměnné a bloky kódu, které je možné snadno použít na různých místech projektu. Tím pádem se snižuje riziko vzniku chyb a zjednodušuje se proces úprav v kódu (An Introduction to SASS CSS: The CSS Pre-Processor, 2009).

Pro snadnější pochopení je dobré si ukázat zápis například proměnné a loopu v SASSu.

```scss
$vaseBarva: #000058;
.vasElement {
  color: $vaseBarva;
}
```

*Obrázek 3 - Zápis proměnné v SCSS*

```scss
@for $vasePromenna from 35px to 85px {
  .margin-#{$vasePromenna} {
    margin: $vasePromenna 10px;
  }
}
```

*Obrázek 4 - Zápis smyčky v SCSS*

#### 3.1.3.2 LESS

LESS, uvedený na trh v roce 2009 Alexisem Sellierem, byl zformován pod vlivem Sass a adoptoval mnohé jeho rysy, jako jsou směsi, proměnné a vkládání prvků. Později však LESS ovlivnil Sass, když nová syntaxe SCSS čerpala inspiraci z LESS. Preprocesor LESS, který je knihovnou v jazyce JavaScript, rozšiřuje základní možnosti CSS. Vzhledem k tomu, že je napsán v JavaScriptu, můžete jej spouštět jako balíček npm nebo kompilovat LESS přímo v prohlížeči přidáním souborů .less a převodníku LESS do hlavičky vaší HTML stránky (Monus, 2023).

Preprocesor LESS obsahuje i speciální chráněné mixiny, které umožňují použití elementární podmíněné logiky v LESS. Toto je klíčové, jelikož LESS neposkytuje tak sofistikované podmíněné operace jako Sass (například nemá příkaz if-else), přesto však můžete s pomocí těchto mixinů chráněním docílit významných výsledků (Monus, 2023).
I zde je příklad deklarování a následné použití proměnné v LESSu:

```
@primarni-barva: seashell;
@primarni-barva-pozadi: darkslategrey;

body {
    color: @primarni-barva;
    background: @primarni-barva-pozadi;
}
```

*Obrázek 5 - Zápis proměnné v LESS CSS*

### 3.1.4 JavaScript

JavaScript stál u zrodu proměny internetu z původních statických stránek na současný interaktivní web. Zatímco HTML určuje, co je na stránkách obsahem a CSS jak stránky vypadají, JavaScript dává webu "život" tím, že určuje jeho chování. Je to právě JavaScript, který přináší dynamiku a interaktivitu na webové stránky, protože umí zasahovat jak do obsahu definovaného v HTML, tak do vzhledu stanoveného v CSS. Díky tomu má JavaScript v rukou klíče k moci nad webovými technologiemi (Ranjan, 2020, s. 9).
JavaScript dominuje na trhu webových technologií díky široké nabídce silných frameworků a knihoven, které usnadňují a systematizují vývoj webů. Na webu se s JavaScriptem setkáme všude, nejen v kódu běžícím na straně klienta, ale i v logice na serveru. Díky bohatě vybaveným knihovnám a frameworkům jako Angular nebo React mohou vývojáři přidávat nové funkce bez nutnosti psát komplexní kód. JavaScriptové frameworky zjednodušují složité příkazy na jednoduché bloky kódu, což programování značně usnadňuje a urychluje (Ranjan, 2020, s. 9).
V současnosti spočívá vývoj softwaru hlavně ve skládání: rozdělení komplexních úkolů na jednodušší a jejich postupné řešení, které se nakonec spojí do finální podoby aplikace. Moderní vývojové frameworky podporují tento kompoziční přístup, kdy je celkový design aplikace rozčleněn do menších částí. Tyto části se pak krok za krokem vytvářejí a skládají do finálního celku. Seznam JavaScriptových frameworků neustále roste a rozvíjí, čímž ještě více zvyšuje jejich význam v online prostředí (Ranjan, 2020, s. 9).

## 3.2 CSS frameworky

V oblasti front-endu obvykle CSS framework poskytuje soubory a složky, které obsahují kód pro HTML, CSS a JavaScript. Každý framework má podrobnou dokumentaci, vysvětlující jeho účel, obsažené komponenty a způsob jejich využití. Použití CSS frameworků výrazně usnadňuje práci vývojářům, kteří by jinak museli vynaložit mnoho úsilí na vytvoření layoutů, grid systémů, UI komponent a dodržování nejlepších praxí. Frameworky rovněž řeší dvě základní problémové oblasti front-end vývoje – kompatibilitu s prohlížeči a tvorbu responzivních designů. Většina CSS frameworků byla navržena tak, aby podporovala různé prohlížeče a platformy, od mobilních zařízení po desktopové počítače (Canziba, 2018, s. 293).

Podle Bose (2011) má používání CSS frameworků tyto výhody:

- CSS frameworky umožňují vývojářům a designérům začlenit do webových stránek rozmanité pokročilé funkce a designové prvky, jako jsou formuláře, tlačítka, navigační panely či drobečková navigace, a zároveň vytvářet přehledné a symetrické layouty.
- Tyto frameworky zjednodušují vývoj webových stránek, které jsou kompatibilní s různými prohlížeči, a snižují tak riziko chyb během testování v různých prohlížečích.
- Vzhledem k tomu, že frameworky nabízí již hotové styly, jejich použití urychluje a usnadňuje vývoj webu.
- Vývojáři mohou díky nim efektivně vytvářet přívětivá a esteticky přitažlivá uživatelská rozhraní, která lze během projektu flexibilně upravovat, aniž by bylo nutné začínat vždy od základů.

### 3.2.1 Bootstrap

Bootstrap je nástroj určený pro tvorbu webových rozhraní, který byl poprvé vytvořen v roce 2011 pracovníky Twitteru, Markem Ottou a Jacobem Thorntonem. Dnes je tento framework dostupný jako open source a patří mezi nejoblíbenější nástroje pro webový vývoj. Díky možnosti bezplatného použití v soukromém, vzdělávacím a komerčním sektoru si Bootstrap rychle získal popularitu a nyní je využíván mnoha organizacemi po celém světě, včetně tak významných jmen jako NASA, Walmart a Bloomberg (Jakobus, 2018, s. 18).

Od prvního uvedení Bootstrapu od Twitteru v srpnu 2011 se událo mnoho změn. Základní verze Bootstrapu poskytla vývojářům soubor CSS pravidel pro organizaci webových

stránek, tvorbu formulářů a tlačítek a podporu v oblasti designu a navigace webu. Bootstrap 4 zůstává ve své jádru věrný svým předchůdcům ve způsobu využití pro tvorbu rozložení a uživatelských rozhraní, která mají jednotný vzhled. Dosahování tohoto jednotného designu je založeno na použití Bootstrap stylů pro různé prvky, jako jsou tlačítka a formuláře (Jakobus, 2018, s. 21).

Ačkoliv zůstávají základní vlastnosti Bootstrapu zachovány, vnitřní struktura frameworku prošla kompletní přeměnou, protože Bootstrap 4 byl kompletně přepsán. Framework tak přináší nové pomocné třídy a komponenty, zatímco ukončuje podporu pro některé a zavádí novou pro jiné. Došlo také ke změnám ve vzhledu jednotlivých komponent a v metodách, jakými by měly být definovány a vytvářeny doplňky od třetích stran. Tým stojící za Bootstrapem 4 rovněž zastavil podporu pro některé starší prohlížeče a začal podporovat nové, například prohlížeč a WebView systému Android v5.0 Lollipop (Jakobus, 2018, s. 21).

Podle Bose (2011) přináší používání Bootstrapu tyto výhody:

- Rozsáhlý ekosystém: Bootstrap se mezi front-end frameworky vyznačuje nepřekonatelným ekosystémem. Nabízí obrovskou knihovnu layoutů, témat, UI prvků, panelů, modálních oken, tlačítek, upozornění, karet atd., ze kterých mohou vývojáři a designéři vybírat a implementovat je do svých projektů. Kromě toho je Bootstrap podpořen špičkovou komunitní podporou v oboru.
- Zrychlený prototyping: Díky Bootstrapu mohou designéři psát svůj HTML kód, použít relevantní CSS třídy a dosáhnout responzivity webu. Nemusí ztrácet čas přizpůsobením pro nekompatibilitu prohlížečů, CSS pozicováním a podobně.
- Podpora od Twitteru: Není překvapením, že když významný komerční subjekt podporuje open-source projekt, uživatelé mohou být ujištěni o jeho stálosti a důvěře v něj mezi lidmi, kteří se v oboru vyznají. Fakt, že Bootstrap vznikl a je podporován společností Twitter, potvrzuje jeho efektivitu.
- Podpora SASS a LESS: Ačkoliv většina vývojářů nepoužívá LESS, na významných projektech se stále spoléhá. Podpora SASS je také velmi žádaná. Ne mnoho CSS frameworků kromě Bootstrapu podporuje obě technologie.

### 3.2.2 Tailwind CSS

Tailwind CSS nabízí rychlý a snadný způsob, jak aplikovat styly na webové stránky. Jedná se o framework, který dává přednost utilitám a umožňuje expresní tvorbu vlastních uživatelských rozhraní. Tailwind je velmi flexibilní a nabízí základní stavební prvky pro tvorbu designu bez nutnosti zásahů do předdefinovaných stylů. Výhodou Tailwindu je, že neukládá žádné konkrétní designové požadavky a nechává na uživatelích, aby skládali komponenty a vytvořili jedinečné rozhraní. Tailwind zpracuje CSS soubor podle konfiguračního souboru a vytvoří výsledný kód (Alif, 2023).

Tailwind CSS se ukazuje jako skvělá volba pro vývojáře díky svému účinnému přístupu zaměřenému na utilitní třídy, který zjednodušuje proces vývoje. Tato metodika umožňuje rychlé sestavování moderních a přizpůsobivých webů bez nutnosti psát rozsáhlé CSS kódy. S širokou škálou utilitních tříd lze jednoduše tvořit rozložení a styly a zvládat interaktivitu s minimální námahou, čímž se zmenšuje potřeba komplexních CSS souborů a zvyšuje se udržitelnost a čistota kódu. Výraznost utilitních tříd Tailwindu také napomáhá lepší orientaci ve funkčnosti tříd a podporuje spolupráci mezi vývojáři (Alif, 2023).

Dále Tailwind zastává flexibilitu a přizpůsobivost. Na rozdíl od jiných frameworků s pevně danými komponenty a styly, Tailwind nabízí flexibilní prostředí pro tvorbu originálních designů s dodržením konzistence a nejlepších praktik. Vývojáři díky tomu mohou vytvářet uživatelská rozhraní přizpůsobená specifickým potřebám jejich projektů. Tailwind také usnadňuje vytváření responzivních webů adaptabilních na různé velikosti displejů. Tailwind CSS tak představuje ideální rovnováhu mezi efektivitou a možností kreativního vyjádření, což jej činí významným nástrojem pro vývoj moderních webových aplikací (Alif, 2023).

Bose (2011) jmenuje následující výhody při používání Tailwindu:

- Vysoká úroveň přizpůsobení: Tailwind CSS umožňuje upravit výchozí nastavení pomocí souboru tailwind.config.js, což zjednodušuje úpravy vzhledu, motivů, rozestupů, barevných schémat atd. Využití funkcí Tailwindu usnadňuje řízení projektu a tvorbu atraktivních webových stránek.
- Integrované utility vzorce: Tailwind CSS snižuje potřebu vytvářet mnoho jedinečných tříd díky předdefinovaným utility vzorcům pro obvyklé úkoly, jako je uspořádání tříd, jejich kaskádování atd. Výsledkem je jednodušší proces tvorby vlastních komponent, kdy lze hodnoty efektivně odvodit z konfiguračních souborů skrze funkci theme().

- Redukce velikosti souborů s PurgeCSS: PurgeCSS efektivně snižuje velikost CSS odstraňováním nepoužívaných tříd, což je v kombinaci s Tailwind CSS velmi prospěšné. Optimalizace prostřednictvím PurgeCSS zefektivňuje správu CSS, zejména ve fázi přípravy na spuštění projektu.
- Zjednodušená responzivita: Tailwind CSS adoptuje strategii "mobile-first", umožňující aplikovat utility třídy podmíněně pro různé breakpointy, což značně ulehčuje vytváření komplexních responzivních designů přímo v HTML. To umožňuje bezproblémové a efektivní vytváření responzivních webových rozhraní.

### 3.2.3 Foundation

Tento framework pro front-end vývoj nabízí systém mřížky, HTML, SASS, CSS prveků uživatelského rozhraní, šablony a kódy pro různé prvky jako navigaci, tlačítka, typografii a formuláře. Kromě toho zahrnuje i volitelné funkce dostupné díky rozšířením v JavaScriptu. Od roku 2019 je Foundation, původně spravovaný firmou ZURB, udržován jako open-source projekt dobrovolníky. Framework preferuje strategii "mobile-first" a je obzvláště vhodný pro vývoj velkých webových aplikací, které potřebují specifický design (Bose, 2011).

Foundation překračuje běžné požadavky na CSS framework díky své komplexní a modulární kolekci nástrojů, které jsou navrženy tak, aby odpovídaly potřebám většiny vývojářů pracujících na front-endu. Poskytuje specifické komponenty pro tvorbu webových stránek a e-mailů, přičemž každá sada je optimalizována pro použití v daném kontextu. Navíc nabízí rozhraní příkazové řádky (CLI), které je velkým přínosem pro vývojáře využívající nástroje jako Webpack pro správu modulů (Bose, 2011).

Foundation poskytuje vývojářům pracujícím na front-endu plnou kontrolu nad jejich uživatelskými rozhraními. Neomezuje je v používání specifického stylu nebo programovacího jazyka, což z něj dělá oblíbený nástroj mezi širokou škálou vývojářů (Bose, 2011).

Foundation nabízí nejen obvyklé komponenty pro uživatelské rozhraní, ale také zahrnuje funkce jako systém pro responzivní obrázky, tabulku s cenami, responzivní embedování, ověřování formulářů, podporu pro psaní zprava doleva a další. To dává vývojářům mnohem více nástrojů a funkcí, s kterými mohou na svých webových projektech pracovat (Bose, 2011).

### 3.2.4 Bulma

Bulma je framework pro návrh webových stránek s otevřeným zdrojovým kódem, který slouží k vytvoření atraktivních webových stránek. Jako výkonný nástroj pro CSS umožňuje efektivně vytvářet rozšiřitelný kód pro front-end, přičemž klade důraz na přístup "mobilní zařízení na prvním místě". Nabízí integrované CSS komponenty pro vývoj flexibilních webů. Výhodou Bulmy je, že odstraňuje složitost spojenou s programováním od základů tím, že poskytuje uživateli potřebné nástroje a funkce pro pohodlné a efektivní tvorbu interaktivních webů (Shenoy, 2019, s. 1).

Bulma je primárně určena pro mobilní zařízení, ale lze ji s určitými úpravami použít i pro desktopové počítače. Byla vytvořena s ohledem na mobilní platformy, a proto jsou weby vytvořené s pomocí Bulmy v souladu s aktuálními a předpokládanými trendy vyhledávače Google. Je také optimalizovaná pro vyhledávače (SEO) a přizpůsobená budoucím trendům, protože Google preferuje mobilní weby při indexaci oproti desktopovým verzím (Shenoy, 2019, s. 3).

### 3.2.5 Skeleton

Podle Bose (2011) se Skeleton neprezentuje jako typický CSS framework, ale jako "extrémně jednoduchá responzivní základní šablona". Jeho minimalistický charakter je zřejmý – skládá se pouze z 400 řádků zdrojového kódu.

Tento minimalistický nástroj byl vytvořen pro design CSS prvků, které fungují jak na velkých obrazovkách, tak na mobilních zařízeních. Zahrnuje všechny základní prvky pro adaptivní design a organizuje stránku do více 12-sloupcových mřížek s limitem šířky 960px, což odpovídá potřebám malých, středních i velkých displejů. Přizpůsobení maximální šířky je možné provést jednoduchou úpravou CSS. Syntaxe Skeletonu je navržena pro rychlou a snadnou implementaci, což zjednodušuje tvorbu responzivních webů (Bose, 2011).

I zde Bose (2011) jmenuje konkrétní výhody plynoucí z používání CSS frameworku Skeleton:

- Díky své lehkosti je Skeleton snadno uchovatelný, spravovatelný a manipulovatelný.
- Navržen s prioritou mobilních zařízení, tento nástroj je skvělým startovacím bodem pro designéry.
- Obsahující pouze nezbytné komponenty a HTML elementy, včetně podpory pro grid, je Skeleton ideální volbou pro malé projekty.

- Vzhledem k tomu, že zahrnuje základní prvky a komponenty CSS, je jeho učení a používání jednoduché i pro začínající vývojáře.

## 3.3 Responzivní web design

V teoretické části je podrobně prozkoumána oblast responzivního designu, který slouží jako základní stavební kámen pro front-endové frameworky. Zjištění a poznatky z této kapitoly jsou následně využívány ve praktické části, kde je na reálných příkladech ukázáno, jak lze principy responzivního designu efektivně implementovat při tvorbě webových aplikací. Tímto způsobem je poskytnut komplexní pohled na to, jak teorie responzivního designu ovlivňuje a formuje praktický vývoj webových řešení.

Friedman (2006) představuje responzivní web design jako metodiku podle které by se design a vývoj měly přizpůsobovat uživatelovým potřebám a podmínkám založeným na velikosti obrazovky, používané platformě a orientaci zařízení.

Metodika zahrnuje použití flexibilních mřížek, layoutů, obrázků a CSS mediálních dotazů, které umožňují webu automaticky se adaptovat, například při přechodu z počítače na tablet. Webové stránky by měly být schopné automaticky se upravit, aby vyhovovaly rozdílným požadavkům na rozlišení, velikost obrázků a možnosti skriptování. Navíc by měly být kompatibilní s uživatelskými nastaveními na různých zařízeních, jako je například použití VPN na iPadu, aby nedocházelo k omezení přístupu k obsahu. Webová stránka by měla být vybavena technologií, která se automaticky přizpůsobuje uživatelovým preferencím, což by odstranilo nutnost vytvářet odlišné designy pro každý typ zařízení na trhu (Friedman, 2006).

Marcotte (2010) popisuje *„Nedávno se začala rozvíjet nová disciplína nazvaná „responzivní architektura“, která se zabývá otázkou, jak mohou fyzické prostory reagovat na přítomnost lidí, kteří jimi procházejí. Prostřednictvím kombinace vestavěné robotiky a pružných materiálů experimentují architekti s uměleckými instalacemi a stěnovými konstrukcemi, které se ohýbají, pruží a rozšiřují, jak se k nim lidé přibližují. Pohybové senzory mohou být spárovány s systémy klimatizace, aby upravovaly teplotu místnosti a ambientní osvětlení, jak se místnost naplňuje lidmi. Firmy již vyvinuly „inteligentní skleněnou technologii“, která se může automaticky stát neprůhlednou, když hustota osob v místnosti dosáhne určitého prahu, čímž jim poskytuje další úroveň soukromí.“*

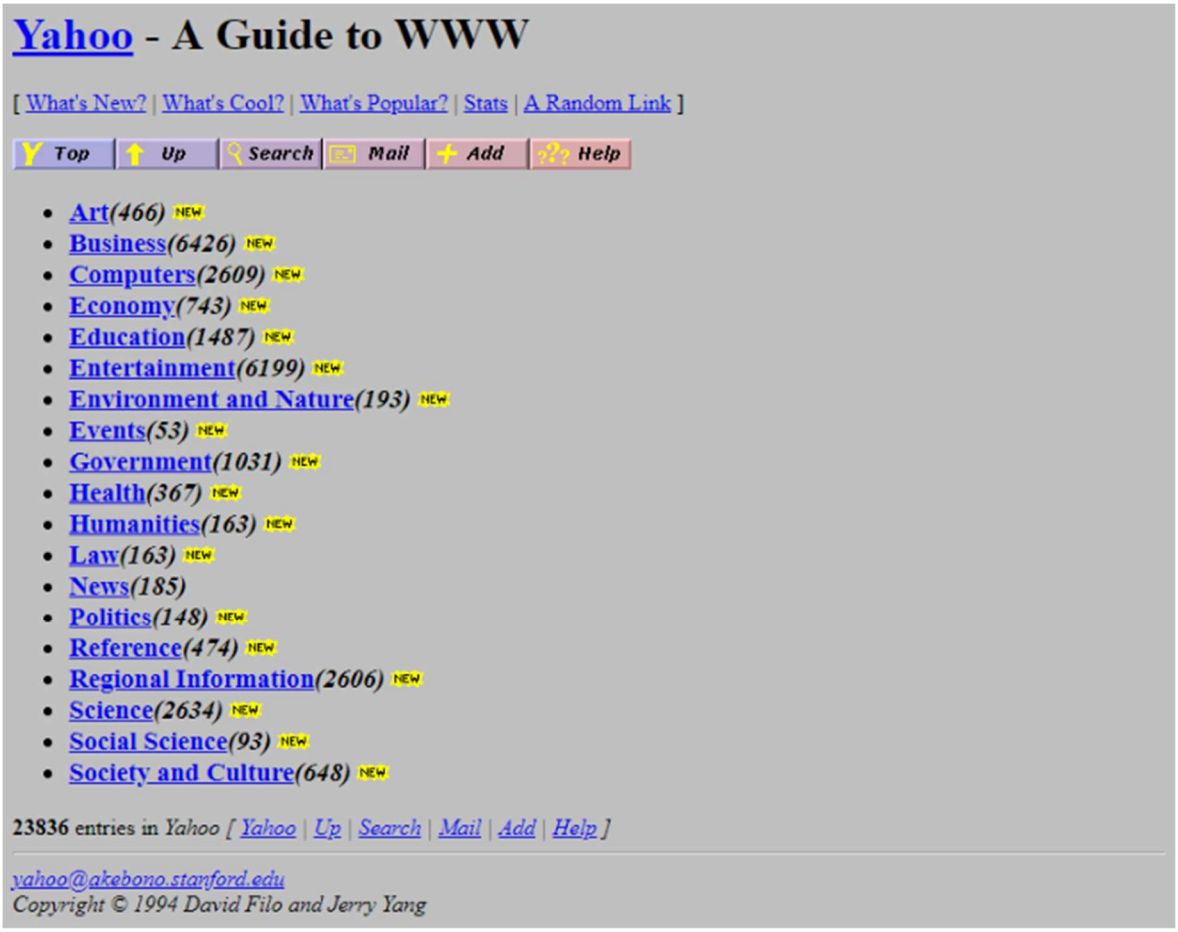
Friedman (2006) reaguje na slova Marcotte (2010) a dodává, že pokud aplikujeme principy responzivní architektury na webdesign, získáme sice podobný, ale zároveň zcela nový přístup. Proč tedy vytvářet specifický design pro každou skupinu uživatelů, když ani

architekti nedesignují budovu pro každou skupinu lidí, která ji použije? Stejně jako responzivní architektura by se měl i design webu automaticky přizpůsobovat a neměl by vyžadovat neustálé vytváření specifických řešení pro různé kategorie uživatelů.

### 3.3.1 Historie

První webová stránka byla zpřístupněna 6. srpna 1991 a jejím tvůrcem byl Tim Berners-Lee, který na ní prezentoval projekt World Wide Web (W3). Stránka byla hostována na počítači NeXT v rámci CERN, což je Evropská organizace pro jaderný výzkum. I když původní stránka již není v provozu, CERN v roce 2013 inicioval projekt, jehož cílem bylo „uchovat digitální aktiva, která jsou spojena s narozením webu.“ Byly obnoveny původní názvy strojů, IP adresy a URL adresa první webové stránky, a to v nejlepší možné míře (Koishigawa, 2021).

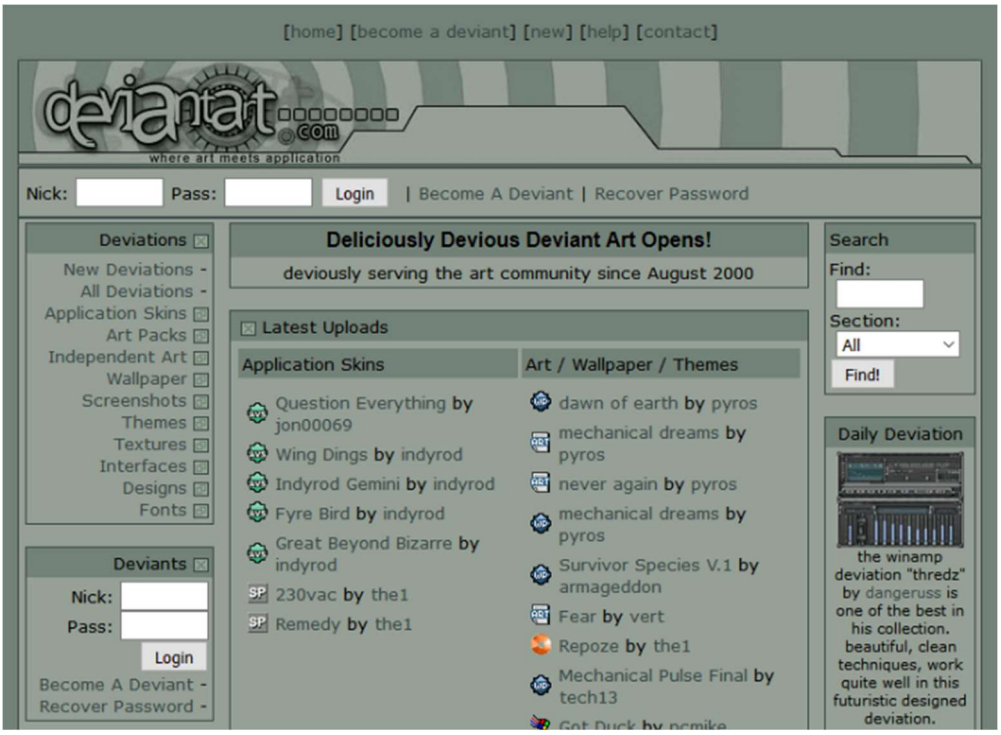
Od doby, kdy Berners-Lee uvedl na internet první web, prošel web rychlým vývojem. Každý rok byly spuštěny tisíce nových webových stránek a vývoj nových designových technik držel krok s rychlým rozvojem webových technologií. V raných 90. letech byl design webů velice základní. Většina webů využívala jednoduchých struktur s hlavičkami, odstavci a prvotními seznamovými značkami jako <dl>, <dt> a <dd> pro uspořádání informací (Koishigawa, 2021).




*Obrázek 6 - Jednoduchý příklad Web Designu (Yahoo!, 1994)*

Složitější webové stránky se musely spoléhat na tabulky, aby mohly správně uspořádat obsah stránky a vytvořit prvky jako je navigace a postranní lišty, které jsou dnes standardem. Přestože už nějaké metody pro stylizaci webových stránek existovaly, CSS bylo poprvé navrženo Håkonem Wium Liem v roce 1994 během jeho působení v CERNu. Následně v roce 1996 představilo World Wide Web Consortium (W3C), založené Berners-Leem, první oficiální specifikaci CSS1 pro kaskádové styly. Díky CSS a technologiím jako JavaScript a Flash se weboví vývojáři mohli stát více inovativními a experimentálními ve svých designech (Koishigawa, 2021).

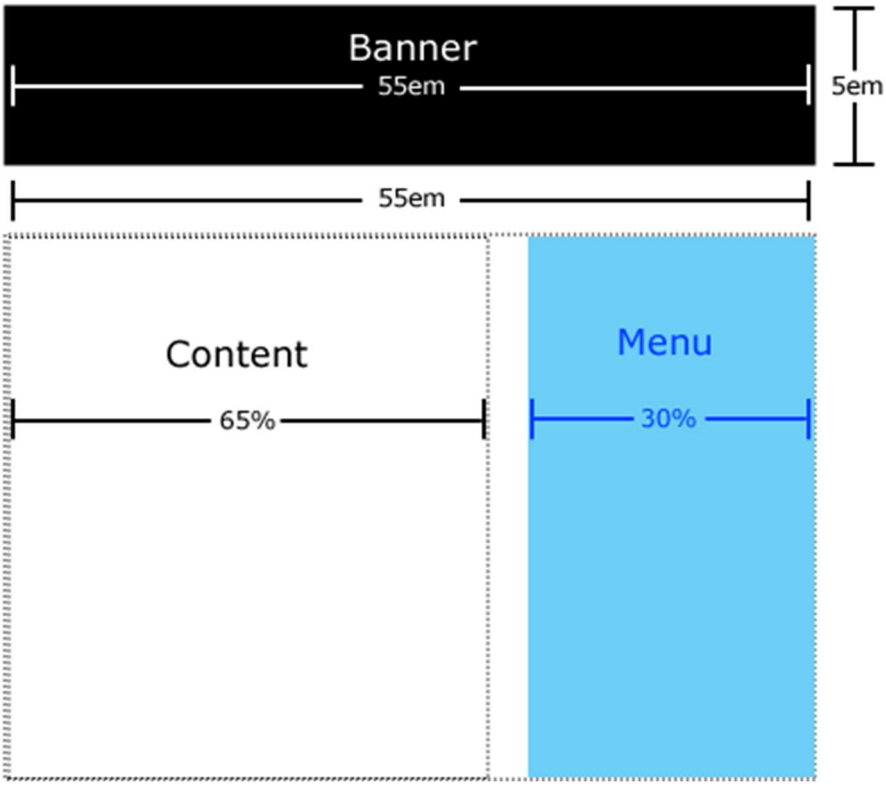
Jak Koishigawa (2021) dále uvádí. V pozdních 90. letech a na začátku nového tisíciletí se vyvinuly vzorce v oblasti webdesignu a uživatelského rozhraní, což vedlo k tomu, že webové stránky začaly vypadat podobně jako ty, které používáme v současnosti.




*Obrázek 7 - Web Design v 90. letech (DeviantArt, 2000)*

Po rozšíření používání CSS se vývojářům začaly vyžadovat značné úsilí a čas na práci s rozvržením stránky, designem a typografií. Nicméně, nemuseli se zatím zabývat adaptací webů na různé velikosti obrazovek, protože v té době používala většina lidí monitory s rozlišením 640x480, 800x600 nebo 1024x768. Přesto se vývojáři setkávali s různými přístupy k práci s těmito rozměry monitorů či prohlížečů, což postupně přispělo k rozvoji responzivního designu, jak ho známe nyní (Koishigawa, 2021).

Koishigawa (2021) dále zmiňuje liquid (tekuté) layouty, které poprvé definoval a zpopularizoval Glenn Davis, byly v té době revoluční a lze je považovat za jednu z prvních hlavních metod responzivního webdesignu. Na rozdíl od layoutů s pevnou šířkou, které mohly na různých monitorech vypadat rozbitě, pokud nesouhlasilo rozlišení, byly tekuté layouty podstatně adaptabilnější. Tyto layouty se dokázaly přizpůsobit různým rozlišením monitorů a velikostem okna prohlížeče. Avšak tekuté layouty nebyly bez chyb. Na stránkách využívajících tento typ rozložení mohl obsah přetékat a text se na menších displejích špatně zobrazovat, zatímco na větších displejích mohlo vznikat příliš mnoho nevyužitého bílého místa.




*Obrázek 8 - Příklad liquid designu (Koishigawa, 2021)*

V roce 2004 blogový článek Camerona Adamse představil novou metodu používání JavaScriptu k výměně stylů v závislosti na velikosti prohlížečového okna, což bylo pojmenováno jako layouty závislé na rozlišení. Tyto layouty vyžadovaly od vývojářů větší úsilí, ale poskytovaly podrobnější kontrolu nad designem webu. Layout závislý na rozlišení fungoval podobně jako raná verze CSS breakpointů, ještě předtím, než se staly standardem. Hlavní nevýhodou byla potřeba vytvářet různé styly pro jednotlivá rozlišení a zajišťovat kompatibilitu JavaScriptu ve všech prohlížečích. S rostoucím počtem prohlížečů, které bylo třeba zohlednit, se jQuery stávalo stále oblíbenějším nástrojem pro překlenutí rozdílů mezi prohlížeči (Carter, 2022).

V době, kdy se množství mobilních zařízení připojujících se k internetu rapidně zvyšovalo, se začaly objevovat koncepty designů přizpůsobujících se rozlišení. Společnosti vyvíjely prohlížeče pro své smartphony, což přineslo vývojářům novou výzvu zohlednit tyto zařízení. Mobilní subdomény sice měly poskytovat uživatelům na smartphonu stejné funkce jako na desktopu, ale byly to úplně oddělené entity. Mít mobilní subdoménu s sebou neslo některé výhody, jako specifické cílení SEO pro mobilní zařízení, což mohlo přilákat více návštěvníků na mobilní verze webů. Přesto však vývojáři museli spravovat dvě varianty téže stránky. V době, kdy Apple právě uváděl na trh svůj první iPad, mnoho webdesignérů stále

využívalo tuto zastaralou metodu pro zpřístupnění webů na různých zařízeních. Na přelomu tisíciletí se vývojáři spoléhali na různé triky, aby usnadnili přístup k mobilním webům, například používali trik s nastavením max-width: 100% pro obrázky. Vše se však začalo měnit, když Ethan Marcotte představil termín "Responzivní webdesign" v článku na A List Apart, což zdůraznilo význam architektonických principů webdesignu, jak je prozkoumal John Allsopp, a položilo základ pro webové stránky, které mohou efektivně fungovat na libovolném zařízení (Carter, 2022).

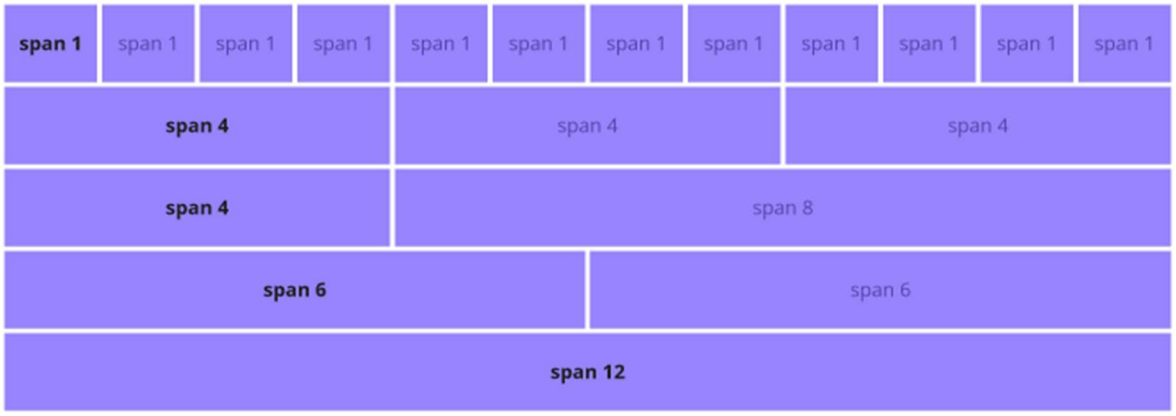
### 3.3.2 Flexibilní struktury (Fluid Grids)

V tekutém (fluidním) designu webu jsou rozměry prvků na stránce definovány v poměru k šířce obrazovky či prohlížečového okna, což znamená, že web se přizpůsobí a změní svou velikost podle aktuální šířky zobrazení. Tento přístup pomáhá zlepšit funkčnost webových stránek na různých zařízeních s odlišnými velikostmi displejů. Fluidní design zaručuje, že layout webové stránky zůstává konzistentní bez ohledu na velikost obrazovky, což přináší výhody pro uživatelské prostředí a zajišťuje, že web bude použitelný pro co nejširší spektrum uživatelů (Juviler, 2024).

Pro snadnější pochopení je možné si uvést příklad, který demonstruje použití plovoucí mřížky (Fluid Grid). Myšlenka plovoucí mřížky spočívá v tom, že se daná webová stránka rozdělí do sloupců a řádků. Sloupců, na které se může webová stránka rozdělit, bývá ve většině případů 12. Šířka těchto sloupců potom je fluidní. Sloupce se poté automaticky přizpůsobují šířce obrazovky, na které je webová stránka zobrazena. Například:

- Na mobilních telefonech se pole formuláře rozprostřou přes celou šířku obrazovky, čímž využijí všech 12 sloupců.
- Na tabletech, které mají větší obrazovku, může být stejné pole formuláře zmenšeno, takže zaujímá jen část obrazovky a umožňuje umístit vedle sebe další pole, s každým polem zabírajícím polovinu šířky (tj. 6 sloupců).
- Na desktopových počítačích, kde je obrazovka ještě větší, je možné mít pole formuláře vedle sebe, přičemž každé z nich zabírá pouze třetinu šířky stránky (tj. 4 sloupce).

Toto rozložení umožňuje optimální využití prostoru a zlepšuje přístupnost a uživatelskou přívětivost webové stránky na různých zařízeních.




*Obrázek 9 - Mřížkový systém pro různá zařízení (How to Use Bootstrap Grid System in Your Website Design, 2017)*

### 3.3.3 Mediální dotazy (Media Queries)

Mediální dotazy jsou způsob, jak cílit na prohlížeč podle určitých charakteristik, vlastností a uživatelských preferencí, a poté na základě těchto informací aplikovat styly nebo spouštět jiný kód. Snad nejběžnější mediální dotazy na světě jsou ty, které cílí na konkrétní rozsahy zobrazovacích oblastí a aplikují vlastní styly, což dalo vzniknout celému konceptu responzivního designu (Galante, 2020).

Media query se skládá z určení typu média (media type, kde výchozí nastavení je pro všechna média jako 'all') a z podmínek, které definují specifické vlastnosti média (media features) s přiřazenou hodnotou nebo určitým rozsahem hodnot (Michálek, 2013).


*Obrázek 10 - Anatomie mediálního dotazu (Michálek, 2013)*

Níže je příklad kódu, který popisuje zápis mediálního dotazu. Tento konkrétní příklad demonstruje použití daných stylů v případě, že šířka prohlížeče je 600 pixelů a více. Pokud je šířka prohlížeče opravdu větší než 600 pixelů, aplikuje se pravidlo na element, pokud je šířka prohlížeče menší než 600 pixelů, pravidlo aplikováno nebude.


*Obrázek 11 - Příklad zapsání mediálního dotazu*

Michálek (2013) uvadí, že v kontextu responzivního designu je klíčový termín breakpoint, což je specifická hodnota určující, kdy se má změnit styl prvků stránky. Breakpointy jsou chápány jako sada kritických hodnot, které se vztahují k danému webu nebo designovému systému. Například, Bootstrap definuje breakpointy na základě šířky okna následovně (Michálek, 2013):

- extra small (do 767px šířky)
- small (768–991px)
- medium (992–1199px)
- large (1200px a více)

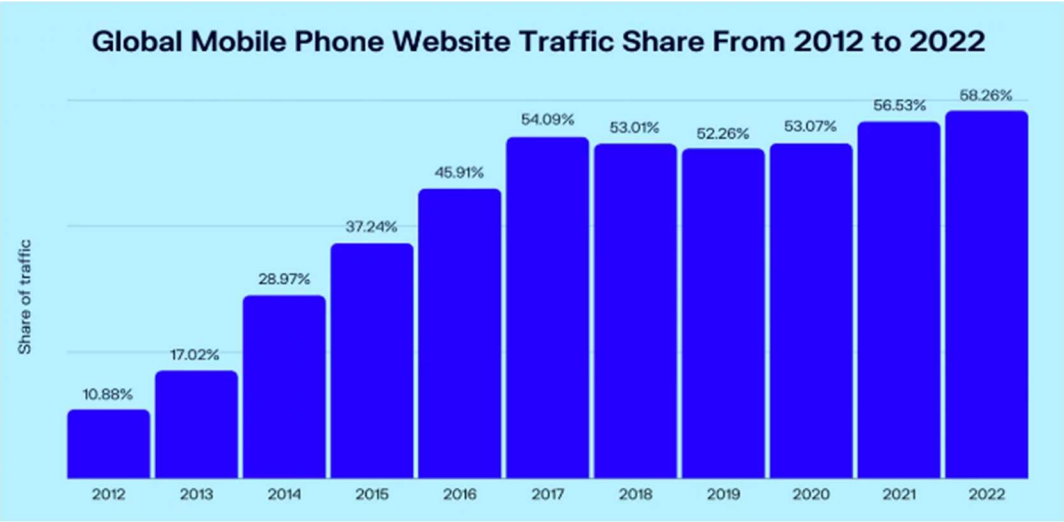
### 3.3.4 Princip Mobile First

Michálek (2014) definuje princip Mobile First jako přístup k designu uživatelských rozhraní, který prioritizuje mobilní zařízení a staví je na stejnou nebo dokonce vyšší úroveň než tradiční stolní počítače s velkými obrazovkami. Tento přístup, který vznikl jako reakce na rostoucí popularitu smartphonů, má navíc další významný pozitivní dopad. Podporuje tvorbu jednodušších, a tím potenciálně efektivnějších a uživatelsky přívětivějších rozhraní.

Maurerová (2023) dokonce mluví o principu Mobile First jako o přístupu, který posunuje responzivní design na novou úroveň tím, že klade důraz na optimalizaci pro mobilní zařízení. V současné době je nezbytné, aby byly webové stránky navrženy s prioritou pro mobily, což znamená začínat design s nejnižším rozlišením pro mobilní zařízení a postupně přecházet k větším rozlišením pro desktopové počítače. Alternativní metodou je začít s designem pro desktop a ten poté upravit pro mobilní zařízení.

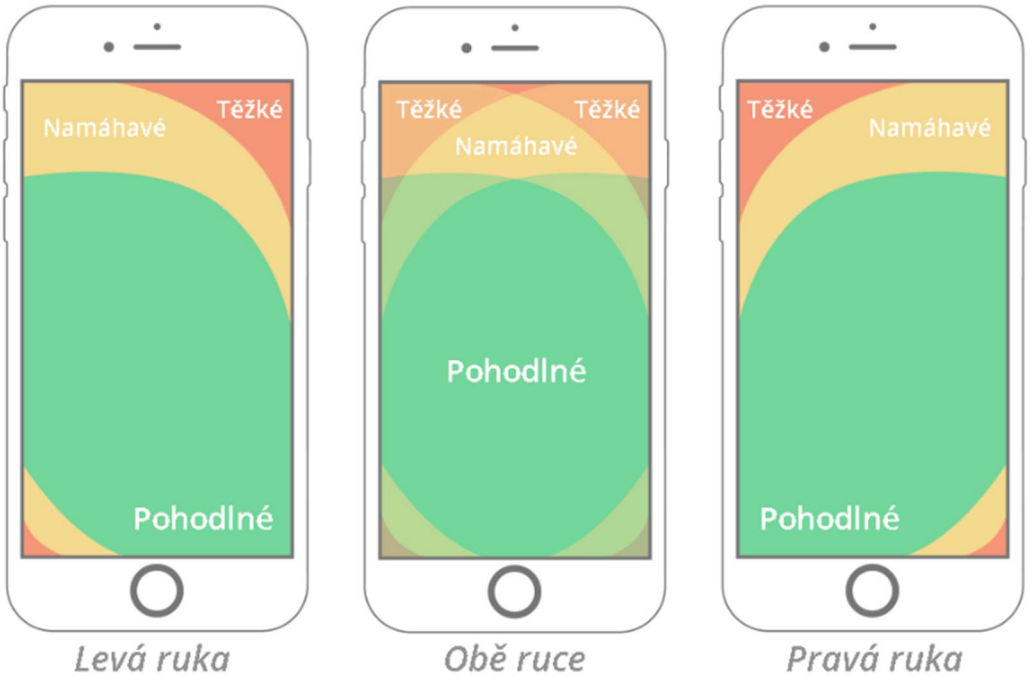
Optimalizace webových stránek pro mobilní zařízení je klíčová pro zvýšení návštěvnosti a uživatelské spokojenosti. Stránky, které nejsou na mobilech snadno použitelné, mají větší šanci, že je uživatel opustí ve prospěch konkurence, což může nepříznivě ovlivnit konverzní poměry, návratnost investiv a celkový image značky (Maurerová, 2023).

Design zaměřený primárně na mobilní zařízení je klíčový pro angažovanost zákazníků. V éře digitální revoluce, která dominovala poslednímu desetiletí, jsou mobilní zařízení na špici změn. Vzhledem k tomu, že zhruba 54 % veškerého webového provozu nyní přichází z mobilů, strategie "mobile first" se stala standardem na trhu (Unadkat, 2011).




*Obrázek 12 - Globální podíl provozu na webových stránkách z mobilních telefonů (Unadkat, 2011)*

Vzhled a uspořádání ovládacích prvků, jako jsou například tlačítka a menu, musí být přizpůsobeny pro snadné ovládání dotykem prstu právě třeba na mobilním zařízení. Měly by být dostatečně velké, aby se na ně dalo klepnout prstem. Doporučuje se 48×48 pixelů. Stejně tak by měly být dostatečně velké mezery mezi tlačítky a ovládacími prvky, aby se návštěvníci webu omylem neproklikli někam jinam. Minimální velikost mezery v tomto případě se doporučuje 8 pixelů (Maurerová, 2023).




*Obrázek 13 - Doporučení pro snadné ovládání dotykem prstu (Mauerová, 2023)*

## 3.4 JavaScript frameworky

### 3.4.1 Angular

Angular, původně vyvinutý Googlem, je rozsáhlý nástroj pro vývoj webových aplikací založený na MVVM vzoru. O šest let později došlo k rozštěpení na dvě linie - starší Angular 1.x, který navazuje na původní AngularJS, a nový Angular 2 založený na moderních ES6 a TypeScript. Obě verze jsou zaměřeny na práci s direktivami a komponentami a mohou se spolehnout na podporu velkých komunit vývojářů a množství rozšíření od externích přispěvatelů (Frisbie, 2023).

Jedná se tedy o vývojářskou platformu založenou na jazyku TypeScript, která obsahuje (What is Angular?, 2023):

- Framework založený na komponentách pro tvorbu rozšiřitelných webových aplikací
- Řadu knihoven, které jsou navzájem dobře propojené a nabízejí mnoho funkcionalit, jako je navigace v aplikaci, řízení formulářů, komunikace mezi klientem a serverem a mnoho dalšího
- Kolekci nástrojů pro vývojáře, které usnadňují práci s kódem, jeho sestavení, testování a aktualizace

Angular je platforma, která umožňuje rozvoj projektů od jednotlivých vývojářů až po aplikace vhodné pro velké korporace. Nejvíce vyniká díky svému ekosystému, který tvoří

více než 1,7 milionu různorodých vývojářů, autorů knihoven a tvůrců obsahu (What is Angular?, 2023).

### 3.4.2 Vue.js

Harber (2023, s. 31) uvádí, že Vue.js je oblíbeným front-endovým rámcem, jehož výhody si cení mnoho velkých i malých firem. Díky své kompaktnosti, intuitivnímu rozhraní, stabilní funkčnosti a jednoduchosti rozšiřování je často volbou před ostatními alternativami. Pomocí Vue.js lze snadno vytvářet přátelská, funkčně i datově bohatá uživatelská rozhraní, která jsou jednoduchá na správu a nasazení.

Vue.js od roku 2014 neustále získává na popularitě. Přestože spousta firem dávala původně přednost Reactu kvůli spojitosti s Facebookem, Vue.js si získal přízeň díky snadnějšímu osvojení a přehlednějšímu uživatelskému rozhraní (Harber, 2023, s. 34).

S podporou předních technologických firem jako Alibaba, Font Awesome, Gitlab a Apple si Vue.js postupně vybudoval pevné postavení mezi vývojáři. V posledních letech se Vue.js stal jedním ze tří nejčastěji používaných JavaScriptových frameworků pro front-end, které upřednostňují velké korporace a podniky (Harber, 2023, s. 34).

### 3.4.3 React

Aplikace Facebook získala obrovskou popularitu, což vedlo k většímu zájmu o nové funkce. React, který byl vyvinut jako řešení Facebooku pro efektivnější práci vývojářů a rychlejší implementaci novinek, se osvědčil natolik, že byl následně uvolněn jako open-source. V současné době je React dobře etablovaná knihovna pro tvorbu front-endových stránek založených na komponentách, jež se těší velké oblibě a má rozsáhlou uživatelskou komunitu (Rippon, 2023, s. 4).

Rippon tvrdí (2023, s. 4), že React získal velkou popularitu nejen díky používání společností jako Meta, Netflix, Uber a Airbnb, ale také díky bohatému ekosystému nástrojů a knihoven podporovaných mnoha vývojáři. Jeho úspěch je částečně dán snadností použití, protože se specializuje na efektivní tvorbu uživatelského rozhraní (UI) pomocí komponent, které lze flexibilně kombinovat a použít v různých aplikacích. Díky svému specifickému zaměření je možné React začlenit i do aplikací s jinými frameworky, protože nepotřebuje ovládat celou aplikaci, ale spokojí se s rolí v rámci frontendu.

React využívá virtuální DOM k efektivnímu zobrazování svých komponent, což je alternativa k tradičnímu DOM, který určuje strukturu webové stránky. Změny v tradičním

DOM mohou být zdlouhavé a mohou zpomalovat interaktivní aplikace. React řeší tyto výkonnostní problémy tím, že pracuje s virtuální verzí DOM v paměti. Před provedením změn v reálném DOM nejprve React vytvoří novou virtuální verzi a porovná ji s aktuální, aby zjistil nezbytné minimální úpravy. Poté se reálný DOM aktualizuje pouze o tyto vyžadované změny, čímž se zvyšuje celková výkonnost aplikace (Rippon, 2023, s. 4).

### 3.4.4 Svelte

Svelte je vyvíjen jako kompilátor psaný v TypeScriptu, jehož cílem je provést co největší množství práce během sestavování aplikace, nikoli až v prohlížeči. Tento framework vytvořil Rich Harris, bývalý zaměstnanec Guardianu a současný grafický editor v The New York Times. Svelte je využíván mnoha firmami, přičemž na jeho oficiálních stránkách jsou jako příklady uvedeny The New York Times, Square, IBM a Chess.com (Humble, 2021).
Svelte, podobně jako řada současných webových frameworků, preferuje uživatelská rozhraní založená na stavech namísto těch založených na událostech pro webové aplikace. Pro aplikace běžící v prohlížeči, které si přejí pracovat tímto způsobem, představuje výzvu skutečnost, že operace s DOM jsou příliš náročné na zdroje pro to, aby bylo možné uchovávat v paměti a alokovat každý DOM objekt podle specifikací API W3C DOM druhé úrovně (Humble, 2021).
Svelte odstraňuje nutnost použití virtuálního DOMu. Když vytváříte komponentu v Sveltu, píšete kód v souboru, který je vlastně HTML soubor s příponou .svelte. Tento soubor je kompilátorem Svelte transformován do abstraktního syntaktického stromu, z něhož je vytvořena JavaScriptová třída pro import do vaší aplikace. Svelte poskytuje počáteční stav přímo v kódu, ne jako DOM objekty, což způsobuje, že je načítání rychlejší a efektivnější. Následně sleduje změny v proměnných komponent a přímo aktualizuje jen ty části DOM, které se týkají změn, namísto celkového překreslení komponenty (Humble, 2021).
Aplikace vytvořené ve Svelte obvykle potřebují méně kódu pro svou implementaci. Přestože použití počtu řádků kódu jako metriky pro měření produktivity vývojářů není ideální, Harris argumentuje, že projekty s menším množstvím kódu mají obvykle méně chyb, za předpokladu, že je kód sám o sobě dobře čitelný (Humble, 2021).

### 3.4.5 Ember

Podle Fehevari (2023) je Ember JS open-source rámec pro JavaScript, který se zaměřuje na zvýšení produktivity a využívá model MVVM (Model-View-ViewModel). Pro vývoj

využívá HTML a CSS. Jeho oblíbenost mezi vývojáři plyne ze stability a snadné údržby při vytváření jednostránkových a dynamických klientských aplikací. Společnosti jako LinkedIn a Apple používají Ember JS pro své aplikace.

Panchal (2021) uvádí následující výhody při používání Ember JS:

- Ekosystém balíčků v Emberu je rozsáhlý a dobře strukturovaný, což umožňuje snadno najít odpovídající balíčky pro řešení vašich specifických potřeb. Kromě toho Ember poskytuje speciální web pro správu těchto balíčků, kde jsou jednotlivé balíčky systematicky řazené, doplněné o detailní dokumentaci a hodnocení uživatelů. Používání běžných NPM balíčků je rovněž bez problémů.
- Rozhraní příkazové řádky Emberu, určené pro náročné webové aplikace, umožňuje komunikovat s operačním systémem a poskytnout strukturu projektu s užitečnými doplňky, což zaručuje rychlé znovu sestavení projektu a živé obnovení. Toto rozhraní je klíčovým nástrojem pro generování kódu, provedení živých rekompilací a spuštění testů přímo v prohlížeči. Použití nástroje CLI, který je běžný u mnoha uznávaných frameworků včetně Emberu, dokazuje jeho flexibilitu.
- Když byl vydán Ember 2.0, byla představena myšlenka stability bez stagnace. To znamená, že i přes cíl neustálého pokroku by neměl být nikdo opomenut. Ember je proto zpětně kompatibilní, což znamená, že žádná aktualizace nerozbije aplikace běžící na starších verzích. Proces aktualizace proběhne hladce a uživatel bude dostatečně informován o jakékoli nadcházející změně.
- Komunita je aspekt, který se často zmiňuje u mnoha známých frameworků, a Ember JS není výjimkou. S přibližně 900 přispěvateli na Githubu a více než 24 000 dotazy na Stackoverflow je zajištěno, že jakýkoli problém, se kterým se setkáme, byl již v minulosti někým řešen. Diskuze probíhající v rámci komunity jsou zcela transparentní a otevřené veřejnosti. Navíc knihovna obsahuje i rozsáhlou dokumentaci.
- Tým Emberu má v plánu každoročně vydávat novou verzi Emberu, přičemž každé vydání se bude soustředit na specifické téma. Jednou z takových verzí je Ember Octane, která se zaměřuje na výkon a efektivitu. Dle oficiálního webu je hlavním cílem Emberu Octane dělat více s menším množstvím zdrojů, což naznačuje, že učení Emberu bude snazší a samotný framework bude štíhlejší a rychlejší. V Emberu Octane se objeví nativní JavaScriptové třídy,

inkrementální rendering, optimalizace tree-shaking, technika rehydratace a podobné funkce.

## 3.5 Vícekriteriální analýza variant

Popisované javascriptové a CSS frameworky budou v praktické části porovnány a analyzovány. Následující rešeršní část popisuje aplikované metody použité analýzy.

Šubrt uvádí (2011, s. 162), že se teorie a modely vícekriteriálního hodnocení zaměřují na proces výběru nejlepší možnosti z řady přijatelných alternativ a jejich doporučení pro realizaci. Při výběru by měla být uplatněna co největší objektivita, přestože se mohou použít různé postupy a metody pro analýzu těchto variant. Někdy může dojít ke střetu zájmů mezi zadavatelem a analytikem, který varianty hodnotí. Analytik má tendenci k maximální objektivitě, protože nemá obvykle osobní zájem na výsledku rozhodování. Jedním z možných problémů je, že analytik nemusí být plně informován o všech aspektech úkolu, což může vést k doporučení varianty považované za "nejlepší" na základě dostupných informací, i když by jiná možnost, která skončila na druhém místě, mohla být ve skutečnosti vhodnější, zejména pokud jsou rozdíly v hodnocení podle kritérií minimální.

Cílem aplikace metod vícekriteriálního hodnocení v rámci diplomové práce je identifikovat nejlepší a nejpříhodnější možnost mezi front-endovými frameworky, přičemž se bere v úvahu všechny klíčové charakteristiky těchto nástrojů.

$$
\begin{array}{c} \\ a_1 \\ a_2 \\ \vdots \\ a_m \end{array}
\begin{array}{c}
\begin{array}{cccc} f_1 & f_2 & \cdots & f_n \end{array} \\
\begin{pmatrix}
y_{11} & y_{12} & \cdots & y_{1n} \\
y_{21} & y_{22} & \cdots & y_{2n} \\
\cdots & \cdots & \cdots & \cdots \\
y_{m1} & y_{m2} & \cdots & y_{mn}
\end{pmatrix}
\end{array}
$$

*Obrázek 14 - Kriteriální matice (Šubrt, 2011, s. 163)*

Z obrázku číslo 3 je zřejmé, že v rámci modelů pro vícekriteriální hodnocení jsou hodnoty, kterými se posuzují různé možnosti podle stanovených kritérií, organizovány do matice

kritérií. Každý prvek této matice, označený jako $y_{ij}$, představuje ohodnocení i-té možnosti vzhledem k j-tému kritériu. V této matici je reprezentována omezená sada možností *m*, a tyto jsou hodnoceny podle *n* stanovených kritérií.

Šubrt (2011, s. 164) dále uvádí, že kritéria pro posuzování mohou být klasifikována na základě jejich charakteru a možnosti měření. Existují maximalizační kritéria, kde nejlepší možnosti jsou ty s nejvyššími hodnotami. Na druhou stranu, minimalizační kritéria vyžadují hledání možností s nejnižšími hodnotami. Dále jsou kritéria rozdělena na kvantitativní, což jsou kritéria s hodnotami, které lze objektivně měřit. Naproti tomu kvalitativní kritéria obsahují hodnoty, které nejsou přímo měřitelné a vyžadují subjektivní posouzení, často s využitím bodových škál pro hodnocení.

### 3.5.1 Saatyho metoda

Jak uvádí Šubrt (2011, s. 174), Saatyho metoda zahrnuje kvantitativní porovnání kritérií mezi sebou, používající devítibodovou škálu, která umožňuje expertům ohodnotit, jak silně preferují jedno kritérium oproti druhému, s hodnotami od 1 pro rovnocenná kritéria, až po 9 pro absolutně preferovaná kritéria, s možností použití mezistupňů pro jemnější gradaci preferencí.

Při provádění párových porovnání kritérií je důležité zaznamenat výsledky do matice, která má stejný počet řádků a sloupců (čtvercová matice). Platí, že $s_{ij} = 1/s_{ji}$. Na hlavní diagonále této matice se nachází hodnoty jedna, což znamená, že každé kritérium je samo o sobě považováno za ekvivalentní (Šubrt, 2011, s. 175).

$$S = \begin{pmatrix} 1 & s_{12} & \cdots & s_{1n} \\ 1/s_{12} & 1 & \cdots & s_{2n} \\ \vdots & \vdots & \vdots & \vdots \\ 1/s_{1k} & 1/s_{12} & \cdots & 1 \end{pmatrix}$$

*Obrázek 15 - Saatyho matice (Šubrt, 2011, s. 175)*

Poté je možné převést odhadnuté váhy na konkrétní číselné hodnoty. Běžně se k tomu používá metoda, která zahrnuje výpočet normalizovaných vah z geometrických průměrů řádků v Saatyho matici. Tento výpočet (viz obrázek níže) se provádí nalezením geometrického průměru, označený jako $b_i$, pak se normalizují s použitím vzorce pro $v_i$ (Houška, 2013).

$$b_i = \sqrt[n]{\prod_{j=1}^{n} s_{ij}} \qquad v_i = \frac{b_i}{\sum_{i=1}^{n} b_i}$$

### 3.5.2 Metoda váženého součtu

Metoda váženého součtu je jedním z přístupů k hodnocení různých možností, která umožňuje sestavit celkové hodnocení pro každou možnost s cílem identifikovat tu nejlepší. Tento postup také umožňuje seřadit ostatní možnost. Pro určení celkové užitečnosti každé možnosti je potřeba porovnat ji s ideální variantou *H*, která má ve všech kategoriích nejlepší možné hodnoty, reprezentované hodnocením $h_j$, a s bazální variantou D, která má naopak nejhorší hodnoty, označené $d_j$. Na základě tohoto srovnání lze potom vytvořit matici možností a kritérií, kde se jednotlivé hodnoty vypočítají dle zadaného vzorce $r_{ij}$ (Šubrt, 2011, s. 186).

$$r_{ij} = \frac{y_{ij} - d_j}{h_j - d_j}$$

Po dokončení matice je možné hodnoty, které byly vypočítány pomocí vzorce $u(a_i)$, uspořádat od nejnižší po nejvyšší.

$$\mathbf{u(a_i) = \sum_{j=1}^{k} v_j r_{ij}}$$

# 4 Vlastní práce

Vlastní práce je charakterizována vytvořením identické webové aplikace s využitím šesti různých front-endových frameworků, rozdělených na tři pro CSS a tři pro JavaScript, přičemž pro každou kombinaci frameworků je vytvořena jedna verze aplikace. Ve druhé části jsou specifická kritéria stanovena a následně jsou frameworky hodnoceny na základě těchto kritérií.

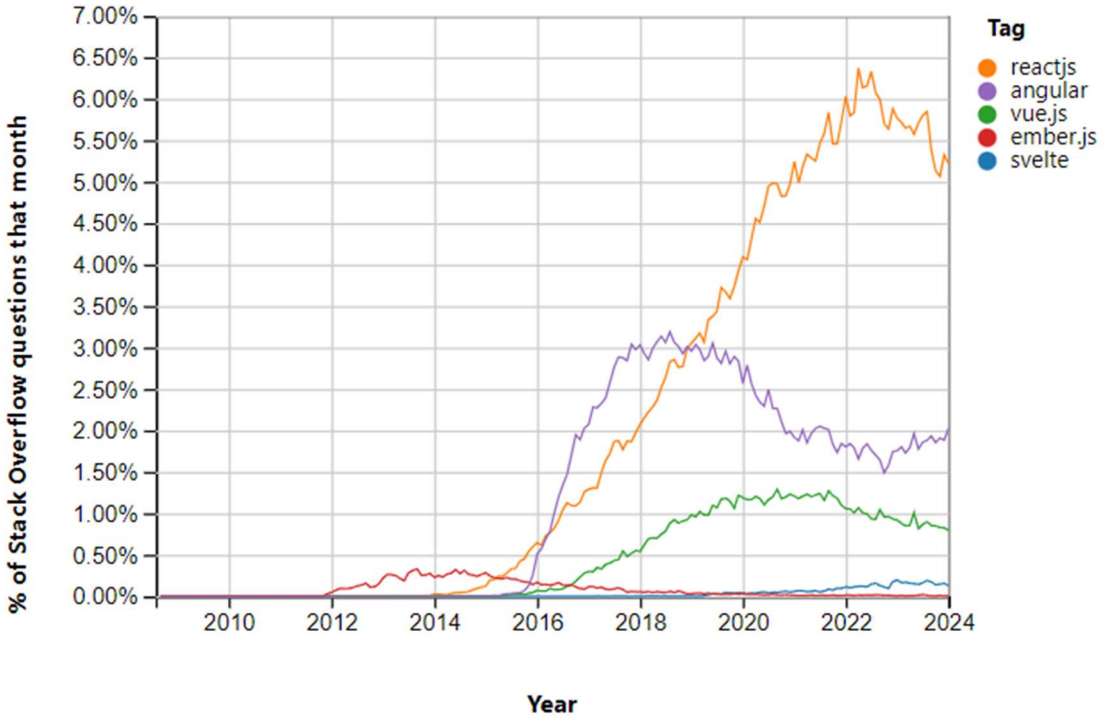
## 4.1 Výběr JavaScript frameworků

Jak již bylo naznačeno v kapitole 2.1, důležitou součástí této práce je porovnání možností, jaké nabízejí jednotlivé JavaScriptové frameworky. Nejprve je nutné ovšem provést výběr frameworků pro pozdější realizaci webových aplikací. Na základě zjištěných poznatků z teoretické části byly vybrány tyto frameworky: Angular, React, Vue, Svelte a Ember. Tyto konkrétní frameworky byly vybrány i následujících důvodů:

- Komponentový přístup – všechny technologie využívají komponentový přístup k vývoji webových aplikací
- Reaktivní a deklarativní UI – všechny frameworky podporují vytváření reaktivních a deklarativních uživatelských rozhraní
- Single-page aplikace – frameworky jsou navrženy pro vývoj single-page aplikací
- Komunita – frameworky mají silný ekosystém a aktivní komunitu
- Vazba dat – frameworky poskytují mechanismy pro vazbu dat mezi uživatelským rozhraním a zdrojem dat

Všechny vybrané frameworky mají své unikátní vlastnosti, funkce, metody, práci s komponentami, ale základní principy, na kterých jsou postaveny, mají mnoho společného. Dalším krokem výběru vhodných JavaScriptových frameworků byly vyřazeny ty frameworky, které měly v posledních letech nízkou podporu a aktivitu u vývojářské komunity. Důležitost aktivního zapojení firmy nebo komunity do vývoje frameworku spočívá především v tom, že aktivní podpora a rozvoj zvyšují šance na včasné odhalení a opravu chyb nebo nedostatků v daném frameworku. S intenzivnějším zapojením komunity do frameworku souvisí také větší počet projektů, které jsou v něm realizovány. V důsledku toho, čím více projektů využívá určitý framework, tím více vývojářů s ním bude mít zkušenosti a dovednosti, což ovlivňuje dostupnost kvalifikovaných pracovníků na trhu. Na

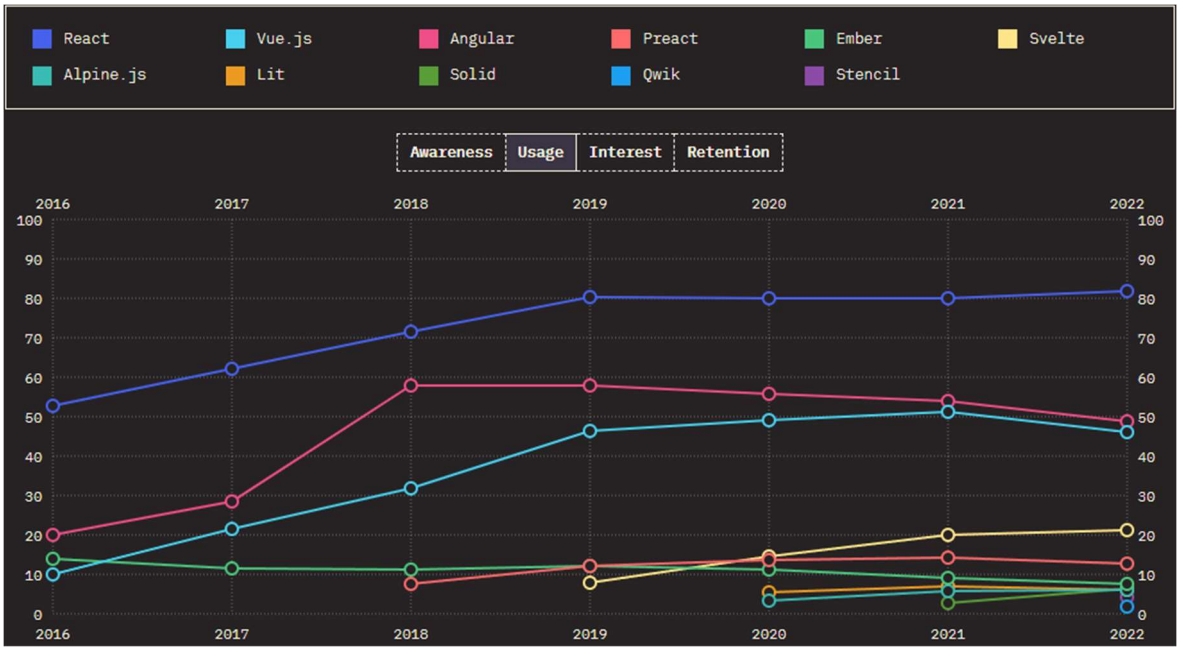
obrázku níže je graf, který porovnává vybrané frameworky na základě dotazů na jednotlivé frameworky na webové stránce stackoverflow.com. Jedná se o komunitu vývojářů, kteří se na této platformě vzdělávají a sdílejí své poznatky ohledně jednotlivých trendů a technologiích.




*Graf 1 - Dotazy na platformě StackOverflow (Stack Overflow Trends, 2024)*

Z tohoto grafu lze vyčíst, že nejvíce dotazů na platformě zaznamenal framework React, následovaným frameworkem Angular. Dále se vývojáři dotazovali na Vue, Svelte a posledním frameworkem v pořadí je Ember.

Další šetření prováděli Sacha Greif a Eric Burel (The 2022 State of JS, 2022), kteří analyzovali v době od 21. listopadu 2021 do 22. prosince 2022 39 472 odpovědí na základě kterých vyplívá, že nejvíce využívaným frameworkem byl koncem roku 2022 React. Na druhém místě Angular. Třetím nejvíce využívaným frameworkem byl Vue, čtvrtým poté Svelte. Ember byl roku 2022 šestým nejvyužívanějším JavaScript frameworkem.




*Graf 2 - Šetření The State of JavaScript (The 2022 State of JS, 2022)*

Podle dat z největšího českého pracovního portálu Jobs.cz byl v období, kdy byla tato práce psána, nejvyšší počet nabídek práce pro projekty využívající Angular a React. To naznačuje, že právě v těchto frameworkách bylo na českém webu realizováno nejvíce projektů. Následující tabulka zobrazuje počet pracovních míst v České republice, kde zaměstnavatelé vyžadují znalosti těchto konkrétních frameworků.

*Tabulka 1 - Počet volných pracovních míst, které vyžadují znalost daného frameworku*

| Angular | React | Vue | Ember | Svelte |
|---|---|---|---|---|
| **50** | **129** | **41** | **0** | **2** |

*(Jobs.cz, 2024)*

Data z portálu freelance.cz, kde programátoři na volné noze mohou zveřejňovat své profily a uvádět v nich, které technologie ovládají, rovněž ukazují, že největší počet programátorů má znalosti ve frameworcích Angular a React. Tyto informace naznačují vysokou popularitu a rozšířené používání těchto technologií mezi nezávislými vývojáři.

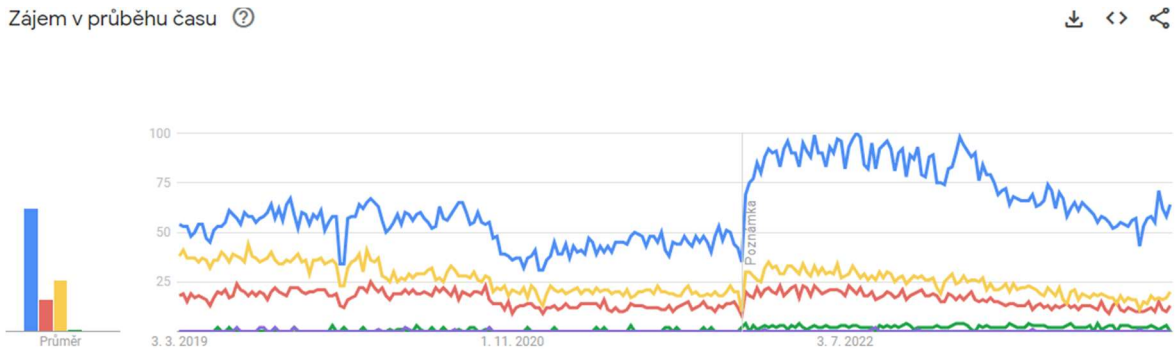
*Tabulka 2 - Počet programátorů na volné noze, kteří mají dovednosti s daným frameworkem*

| Angular | React | Vue | Ember | Svelte |
|---|---|---|---|---|
| **182** | **431** | **105** | **0** | **10** |

*(Freelance.cz, 2024)*

Dalším ukazatelem v jsou trendy vyhledávání pomocí prohlížeče Google Chrome, který je nejpoužívanějším webovým prohlížečem. Graf níže popisuje výsledky vyhledávání

uživatelů prohlížeče Google Chrome. Je zřejmé, že nejvíce výsledků hledání zaznamenal framework React. S větším rozestupem se poté uživatelé nejvíce dotazovali na Angular a Vue. Naopak ve dnech od 18. 2. do 24. 2. 2024 se téměř žádný uživatel nedotazoval na Ember a Svelte (Google Trends, 2024).




*Graf 3 - Výsledky vyhledávání prohlížečem Google Chrome (Google Trends, 2024)*

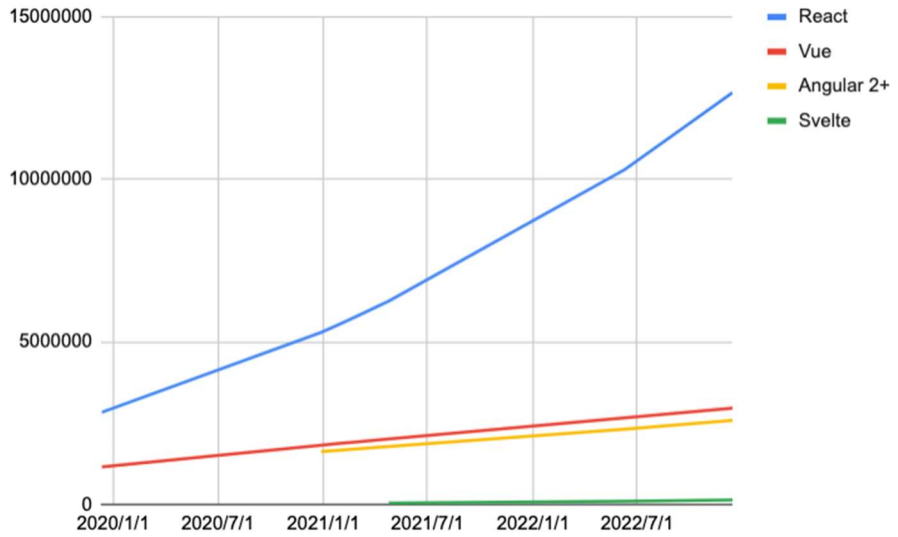

*Obrázek 16 - Výsledky vyhledávání v konkrétním časovém rozmezí (Google Trends, 2024)*

Další pozorování bylo prováděno na základě množství repozitářů na platformě GitHub. Podle Julivera (2006), se jedná o online platformu pro vývoj softwaru. Je využívána pro uložení, monitorování a společnou práci na softwarových projektech. Platforma usnadňuje vývojářům sdílení souborů s kódem a spolupráci s dalšími vývojáři na open-source projektech. GitHub funguje také jako sociální síť pro vývojáře, kde mohou navazovat profesní vztahy, spolupracovat a prezentovat své projekty.

Dále Juliver (2006) uvádí, že od roku 2008, kdy byl GitHub založen, si platforma získala miliony uživatelů a stala se hlavním místem pro spolupráci na softwarových projektech. Nabízí řadu užitečných nástrojů pro sdílení kódu a společnou práci v reálném čase zdarma. GitHub také podporuje uživatele v budování osobního profilu a značky. Můžete navštívit profil jakéhokoli uživatele a zjistit, na jakých projektech se podílí nebo je vede. Díky tomu

se GitHub stává jakousi sociální sítí pro programátory, což podporuje spolupraci na vývoji softwaru a webových aplikací.

Níže již máme graf, který demonstruje počet repozitářů založených na zmíněných technologiích v průběhu posledních let. Nejvíce repozitářů na GitHub má od začátku sledování React. Křivka pro Vue a Angular se kopíruje, ovšem více repozitářů eviduje Vue. Nejméně repozitářů zaznamenal framework Svelte.

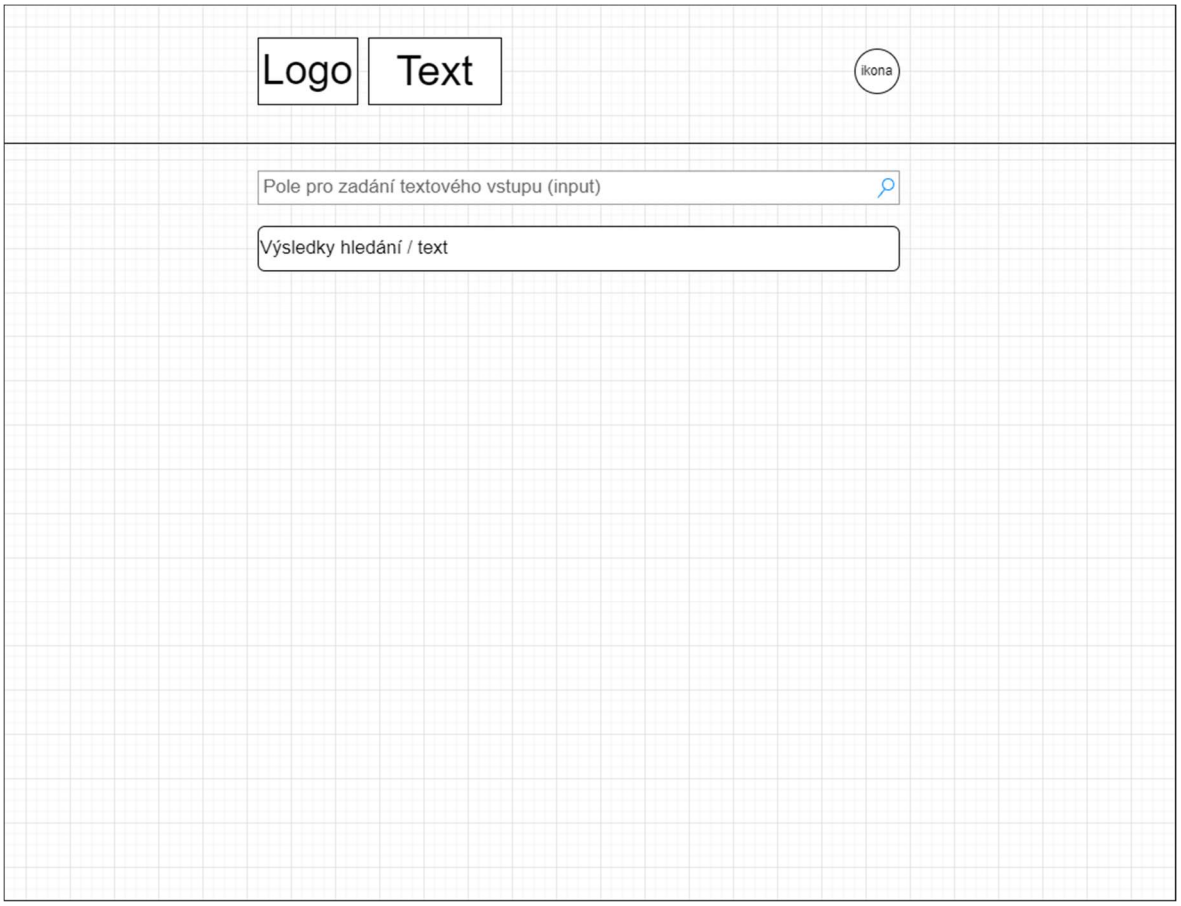



*Graf 4 - Počet repozitářů na GitHub (Krotoff, 2024)*

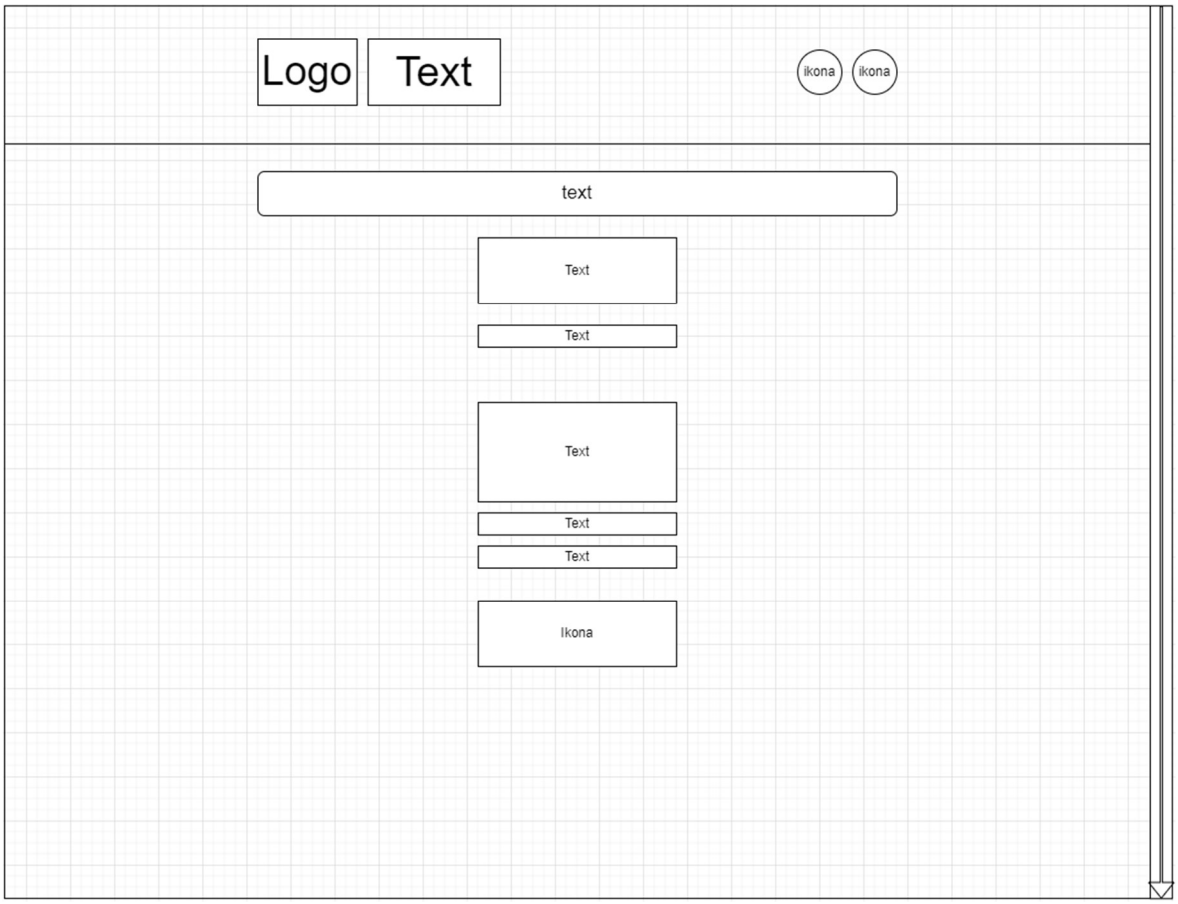
Na základě těchto pozorovaní bylo vyhodnoceno, že nejrozšířenějšími JavaScriptovými frameworky v komunitě vývojářů jsou React, Angular a Vue. Z tohoto důvodu budou v části vývoje webové aplikace použity tyto nástroje, které budou na závěr porovnány na základě stanovených kritérií.

## 4.2 Implementace ukázkové aplikace

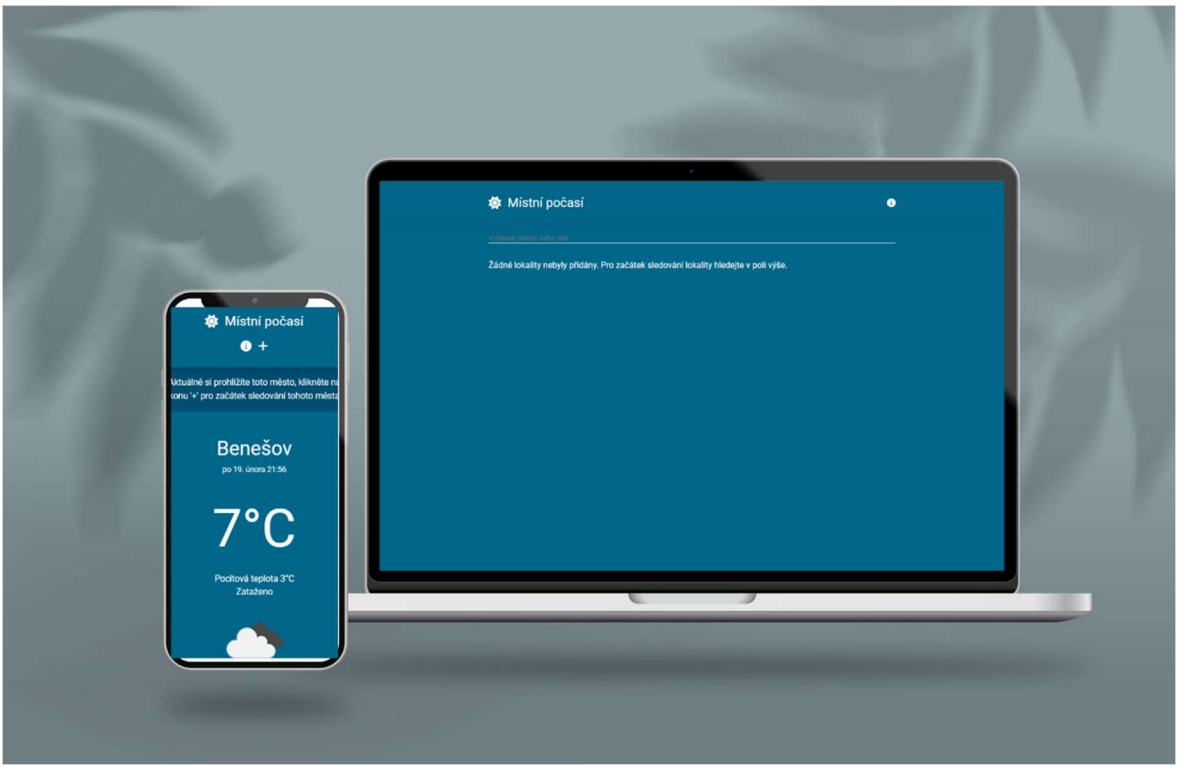
Před samotným začátkem vývoje aplikací byl vytvořen jednoduchý drátový model (wireframe), který slouží jako vizuální vodítko pro rozvržení uživatelského rozhraní. Vizualizaci návrhu uživatelského rozhraní lze nalézt na obrázku číslo 1 a 2. Navigační lišta bude pro všechny náhledy totožná, dynamické bude tělo navigace, které se bude měnit v závislosti na požadavcích uživatele. Na obrázku číslo 3 lze nalézt výsledný produkt vyvíjené webové aplikace.




*Obrázek 17 - Wireframe vytvořené webové aplikace č.1*




*Obrázek 18 - Wireframe vytvářené webové aplikace č.2*




*Obrázek 19 - Vytvořená webová aplikace*

Vzhledem k tomu, že vybrané JS frameworky nabízejí komponentový přístup a jsou navrženy pro aplikace typu Single-page, bude ukázková aplikace využívat komponent a bude se jednat o Single-page aplikaci. I proto je možné vytvořit aplikaci pomocí všech tří vybraných JS frameworků, která bude fungovat na podobných principech. Metody a principy implementované pomocí JS frameworků v ukázkové aplikaci budou popsány v následující kapitole. Samotná logika navržených aplikací bude také velice podobná. Ukázková aplikace by se dala popsat jako standartní aplikace na předpověď počasí. Nyní bude následovat popis logiky aplikace.

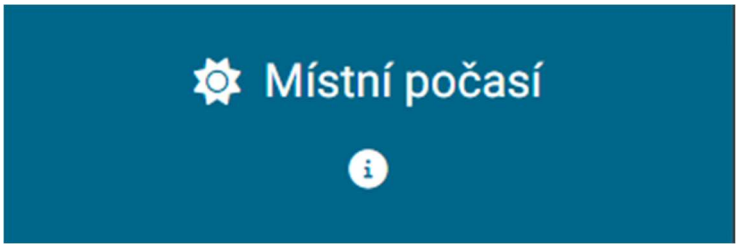
## 4.3 Implementace jednotlivých komponent

### 4.3.1 Komponenta SiteNavigation

Domovská obrazovka aplikace se bude skládat ze dvou komponent. Komponenta SiteNavigation bude připnutá na horní část obrazovky a bude sloužit jako navigační lišta pro uživatele. Tato navigační lišta bude přítomna v celé aplikaci. V levé části lišty bude obsahovat ikonu sluníčka s textem „Místní počasí“. Oba tyto elementy (<i>, <h1>) budou obaleny v nadřazeném elementu ( ) tak, aby bylo možné uživatele přesměrovat na domovskou obrazovku při kliknutí na jakýkoliv element. V pravé části navigační lišty budou dvě ikony. Ikona s písmenem „I“ bude sloužit pro zobrazení modálního okna, které bude poskytovat informace pro uživatele ohledně možností aplikace.

Ikona se znakem „plus“ bude umožňovat přidávat uživatelům města do seznamu „oblíbených měst“. Zobrazení této ikony bude ovšem podmíněno skutečností, zdali se uživatel nachází na stránce, která slouží pro zobrazení informací o počasí v daném městě. Bylo by zbytečné, mít ikonu, pomocí které se přidávají města do seznamu oblíbených měst, když se nenachází na stránce s městem. Pokud uživatel bude přistupovat k aplikace z mobilního zařízení, přesune se levý obsah doprostřed a pravý obsah lišty se přesune taktéž doprostřed, ovšem pod obsah levé části navigační lišty.


*Obrázek 20 - Komponenta SiteNavigation (Desktop)*




*Obrázek 21 - Komponenta SiteNavigation (Mobilní zařízení)*

### 4.3.2 Komponenta HomeView

Domovská obrazovka bude obsahovat také komponentu HomeView. Tato komponenta bude obsahovat pole pro vkládání textu. Na základě vloženého textu se zobrazí seznam měst a států, které budou odpovídat zadanému hledání. Pod textovým polem se budou zobrazovat informace napovídající uživateli. Pokud se v textovém poli nenachází žádný text, vyzve ho aplikace k zadání nějakého vstupu. Pokud zadanému vstupu nebude odpovídat žádný výsledek hledání, vyzve aplikace uživatele k zadání jiného vstupu. V této komponentě se bude vyskytovat podmíněná komponenta CityList, která se bude zobrazovat pouze v případě, že si uživatel již přidal některé město do seznamu „oblíbených měst“. Komponenta CityList bude popsána později. V případě, že uživatel klikne na nějaké město ze seznamu, bude přesměrován na komponentu CityView.

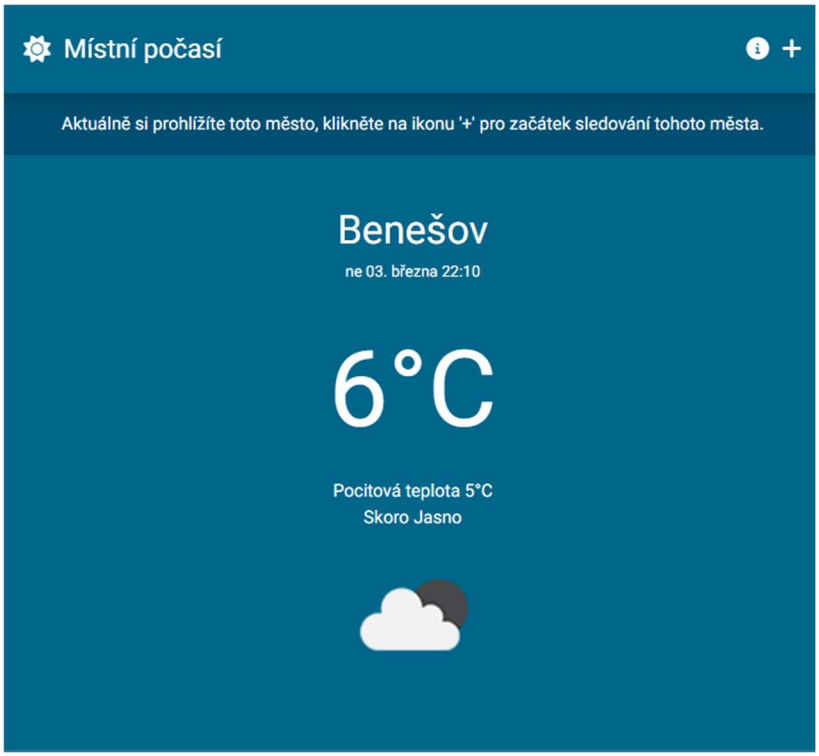

*Obrázek 22 - Komponenta HomeView*

### 4.3.3 Komponenta AsyncCityView

Tato komponenta aplikace poskytuje podrobný přehled o počasí ve vybraném městě, zahrnující aktuální počasí, hodinovou předpověď a 7denní předpověď. Pokud uživatel přistupuje k náhledu města, zobrazí se informativní banner na vrchu stránky, vyzývající k přidání města pro sledování. Pokud si uživatel přidá město do seznamu oblíbených měst, uloží se údaje o městě do localStorage. V lepším případě by se zde měla vyskytovat back-endová část, ovšem pro téma této diplomové práce je toto řešení dostačující.

V sekci s aktuálním počasím je zobrazeno název města, aktuální datum a čas, teplota, pocitová teplota a popis počasí spolu s příslušnou ikonou počasí. Pro hodinovou předpověď je vytvořen posouvací seznam, který ukazuje teplotu a ikonu počasí pro každou nadcházející hodinu. V sekci s 7denní předpovědí jsou uvedeny dny v týdnu s maximálními a

minimálními teplotami a ikonami počasí. Na konci stránky je umístěno tlačítko pro odstranění města z oblíbených, které umožňuje uživatelům snadno spravovat sledovaná města.




*Obrázek 23 - Komponenta AsyncCityView*

### 4.3.4 Komponenta CityCard

Tato komponenta je podmíněnou součástí komponenty HomeView a slouží jako karta města, která zobrazuje základní informace o počasí v daném městě a státu. V levé části karty je název města a státu zobrazený ve flexibilním sloupci pro adaptabilitu na různé velikosti obrazovek. V pravé části jsou teplotní údaje: aktuální teplota zobrazená velkým písmem pro snadnou identifikaci a vedle ní jsou malým písmem uvedeny maximální a minimální teploty, což poskytuje rychlý přehled o očekávaných teplotních výkyvech.


*Obrázek 24 - Komponenta CityCard*

### 4.3.5 Komponenta CityList

Tato komponenta aplikace zobrazuje seznam uložených měst, která uživatel sleduje, s možností kliknout na jednotlivé město pro zobrazení detailnějších informací o počasí. Pokud uživatel nemá uložena žádná města, zobrazí se zpráva vyzývající k jejich přidání.

Pro každé město v seznamu savedCities (localStorage) je generována komponenta CityCard, která zobrazuje základní informace o počasí v daném městě. Kliknutím na kartu města je uživatel přesměrován na stránku s podrobnějším pohledem na počasí v tomto městě.

V případě, že seznam savedCities neobsahuje žádná města, zobrazuje se uživatelům výzva k vyhledání a přidání nových lokalit. Tato interaktivní funkcionalita umožňuje uživatelům snadno spravovat seznam sledovaných měst a přecházet mezi celkovým přehledem a podrobnými informacemi o počasí.

Celkově je tato stránka navržena tak, aby poskytovala uživateli přehledné a interaktivní rozhraní pro sledování počasí ve vybraných městech, s možností snadného přístupu k detailnějším informacím.


*Obrázek 25 - Komponenta CityList*

## 4.4 Charakteristika hodnotících kritérií (JS)

V dalším kroku rozhodování a hodnocení je zásadní stanovit relevantní kritéria. Nejprve jsou tato kritéria obecně definována a poté se specificky uplatňují na každý framework v rámci jednotlivých dílčích kapitol.

Výběr hodnotících kritérií je založen na základě získaných informací studiem odborných zdrojů týkající se javascriptových frameworků a jejich využití při vývoji webové aplikace. Také se zohledňují klíčové faktory při využívání webu, o čemž bylo diskutováno v neformálních debatách s odborníky na webové technologie.

Desai (2023) uvádí že, v roce 2024 je oblíbenost předních front-endových frameworků významným znakem jejich spolehlivosti a obliby mezi vývojáři. Když je framework

oblíbený, obvykle to značí, že má širší skupinu uživatelů, což přináší více aktivní a podporující komunitu.

Dále Desai (2023) uvádí, že důležitým prvkem při hodnocení frameworků jsou nástroje. Udává, že je důležité zohlednit základní funkce a možnosti frontendových frameworků, jako je například směrování, správa stavu a komponenty uživatelského rozhraní, které jsou nezbytné pro efektivní a jednoduchý vývoj moderních webových aplikací. Je nezbytné ověřit, zda framework poskytuje a podporuje funkce jako je připojení k HTTP, práce s formuláři, ověřování požadavků a šablonování.

Singh (2023) zmiňuje výkon jako velice důležité kritérium při porovnávání jednotlivých frameworků. Výkon je jednou z klíčových vlastností každého frameworku. Souvisí se spokojeností uživatele, pokud je framework dobře optimalizovaný, nebude uživatel negativně ovlivněn například rychlostí načítání aplikace.

Je důležité zdůraznit, že výběr kritérií pro hodnocení frameworku má do značné míry subjektivní charakter. To vede k situaci, kde ani členové různých vývojářských komunit nemají jednotný názor na to, který framework je lepší nebo horší a na základě jakých kritérií by měl být hodnocen.

Ze studia odborných zdrojů byl zvolen výkon, jako velice důležitým kritériem pro porovnávání JS frameworků. Proto se autor rozhodl výkon jednotlivých aplikací měřit pomocí nástroje prohlížeče Google Chrome PageSpeed Insights, který umožňuje měřit různé části výkonu webových aplikací. V případě této práce byly zvoleny metriky: Index rychlosti, První vykreslení obsahu, Vykreslení největšího obsahu, Vykreslení obsahu – interaktivita, Velikost frameworku a velikost sestaveného JS souboru. Dalším důležitým kritériem je dokumentace. Při vývoji webové aplikace vývojáři často narážejí na problémy, nebo potřebují vědět, jakou funkci nabízí jimi zvolený framework, která by řešila jejich problém. Právě dokumentace příslušných frameworků je místem, kam se většina vývojářů dívá jako první. S dokumentací souvisí i kvalita popisované práce s komponentami. Právě komponenty jsou stěžejní pro vývoj webových aplikací.

Následující tabulka ukazuje, jaká kritéria byla vybrána pro realizaci vícekriteriálního hodnocení různých možností:

*Tabulka 3 - Kritéria pro vícekriteriální analýzu JS frameworků*

| Číslo | Kritérium | Jednotky |
|---|---|---|
| **1** | Velikost frameworku | MB |
| **2** | Dokumentace | Bodové hodnocení |
| **3** | První vykreslení obsahu | Sekundy |
| **4** | Vykreslení největšího obsahu | Sekundy |
| **5** | Vykreslení obsahu – interaktivita | Milisekundy |
| **6** | Index rychlosti | Sekundy |
| **7** | Velikost sestaveného JS souboru | kB |
| **8** | Komponenty | Bodové hodnocení |

**Velikost frameworku**

Velikost daného frameworku hraje klíčovou roli pro rychlost načítání webové aplikace, protože obecně platí, že větší framework se načítá pomaleji. Velikostí se v tomto smyslu rozumí velikost celého projektu po inicializaci. Inicializační příkazy budou u každého frameworky popsány. Toto kritérium je kvantitativní a má minimalizační povahu. Velikost bude uváděna v megabajtech.

**Dokumentace**

Vzhledem k rozmanitosti a aplikačním možnostem frameworků je klíčové prověřit dokumentaci, kterou poskytují jejich tvůrci. Je důležité, aby dokumentace obsahovala veškeré podstatné informace a instrukce k používání různých komponent. Při hodnocení bude kritická především kvalita, uspořádání a kompletnost poskytnuté dokumentace.

Kritérium dokumentace je kvalitativního charakteru s maximalizační povahou. Tento aspekt nelze kvantitativně změřit, takže jejich hodnocení bude záviset na subjektivním úsudku hodnotitele, který získal praktické zkušenosti během vývoje webové aplikace s použitím testovaných frameworků. Hodnocení variant zde může přijímat numerické hodnoty v rozmezí od 1 do 10, přičemž 1 představuje nejnižší a 10 nejvyšší možné skóre.

**První vykreslení obsahu**

První vykreslení obsahu (First Contentful Paint, FCP) měří, jak dlouho trvá prohlížeči vykreslit první část obsahu DOM po přechodu uživatele na stránku. Obrázky, neprázdné

elementy <canvas> a SVG na stránce jsou považovány za obsah DOM; cokoli uvnitř iframe není zahrnuto (Google, 2024). Nastavení bylo simulováno za podmínek, kdy doba potřebná pro odeslání signálu ze zařízení na server a zpět byla 150 ms. Propustnost sítě byla nastavena na hodnotu 1 638 kilobitů za sekundu. Toto kritérium je kvantitativní a má minimalizační povahu.

**Vykreslení největšího obsahu**

Vykreslení největšího obsahu (Largest Contentful Paint, LCP) měří, kdy je největší obsahový prvek ve viewportu vykreslen na obrazovku. To přibližně odpovídá okamžiku, kdy je hlavní obsah stránky viditelný uživatelům (Google, 2024). Nastavení bylo simulováno za podmínek, kdy doba potřebná pro odeslání signálu ze zařízení na server a zpět byla 150 ms. Dále propustnost sítě byla nastavena na hodnotu 1 638 kilobitů za sekundu. Toto kritérium je kvantitativní a má minimalizační povahu.

**Vykreslení obsahu – interaktivita**

Vykreslení obsahu – interaktivita (Total Blocking Time, TBT) měří celkové množství času, kdy je stránka blokována a nemůže reagovat na uživatelské vstupy, jako jsou kliknutí myší, dotyky obrazovky nebo stisky kláves. Součet se vypočítává přičtením blokující části všech dlouhých úkolů mezi First Contentful Paint a Time to Interactive. Jakýkoliv úkol, který se provádí déle než 50 ms, je považován za dlouhý úkol. Časová část po 50 ms je blokující část (Google, 2024). Nastavení bylo simulováno za podmínek, kdy doba potřebná pro odeslání signálu ze zařízení na server a zpět byla 150 ms. Propustnost sítě byla nastavena na hodnotu 1 638 kilobitů za sekundu. Toto kritérium je kvantitativní a má minimalizační povahu.

**Index rychlosti**

Index rychlosti měří, jak rychle se obsah vizuálně zobrazuje během načítání stránky. Lighthouse nejprve zachytí video načítání stránky v prohlížeči a vypočítá vizuální postup mezi jednotlivými snímky (Google, 2024). Nastavení bylo simulováno za podmínek, kdy doba potřebná pro odeslání signálu ze zařízení na server a zpět byla 150 ms. Propustnost sítě byla nastavena na hodnotu 1 638 kilobitů za sekundu. Toto kritérium je kvantitativní a má minimalizační povahu.

**Velikost sestaveného JS souboru**

V současné době většina zařízení bez problémů zvládá složité webové stránky. Nicméně webové stránky jsou často aktualizovány, někdy i s každou změnou stránky uživatelem. To přináší otázku velikosti webové aplikace, protože ovlivňuje rychlost načítání stránky, rychlost interpretace souborů, uživatelskou zkušenost a síťový provoz. Je žádoucí udržovat

velikost balíčku webové stránky co nejmenší, aby se minimalizoval síťový provoz a zajistila co nejlepší uživatelská zkušenost z hlediska doby načítání. Toto kritérium je kvantitativní a má minimalizační povahu. Velikost bude uváděna v kilobitech.

**Komponenty**

Nástroje zahrnují veškeré integrované aplikace a utility, které zjednodušují vývojářům interakci s frameworkem, jakož i různé pluginy, které mohou rozšířit standardní funkcionalitu daného frameworku. Obvykle jsou frameworky vybaveny CLI (příkazovou řádkou), která umožňuje efektivnější správu různých aspektů vývoje aplikace, jako je vytváření nového projektu, komponenty nebo služby. Toto kritérium je kvalitativní a má maximalizační povahu.

### 4.4.1 Vue

Projekt frameworkem Vue byl inicializován příkazem *npm create vue@latest*. Po této inicializace je uživatel tázán, zdali chce doinstalovat některé knihovny nebo technologie. V případě tohoto projektu byl zapotřebí pouze *Vue Router*. Poté byly pomocí příkazu *npm install* doinstalovány závislosti potřebné pro projekt. V době psaní této práce je stabilní verzí Vue v3.4.21.

**Velikost frameworku**

Velikost frameworku v základní konfiguraci je 107 MB.

**Dokumentace**

Dokumentace Vue je kvalitní zdroj informací pro vývojáře na všech úrovních zkušeností. Poskytuje podrobné průvodce, tutoriály a referenční materiály, které pokrývají základy i pokročilé koncepty Vue. Dokumentace je strukturovaná do hlavních částí, jako například: úvod a instalace, základy, pokročilé koncepty, nástroje, knihovny nebo migrace. Dokumentace je pravidelně aktualizována tak, aby popisovala doporučené postupy a nejnovější změny.

Bodové hodnocení: 10

**První vykreslení obsahu**

První vykreslení obsahu vypočítané pomocí nástroje Statistiky výkonu v prohlížeči Google Chrome pro vývojáře: 2,8 s

**Vykreslení největšího obsahu**

Vykreslení největšího obsahu vypočítáno pomocí nástroje Statistiky výkonu v prohlížeči Google Chrome pro vývojáře: 3,4 s

**Vykreslení obsahu – interaktivita**

Vykreslení obsahu – interaktivita vypočítáno pomocí nástroje Statistiky výkonu v prohlížeči Google Chrome pro vývojáře: 0 ms

**Index rychlosti**

Index rychlosti vypočítán pomocí nástroje Statistiky výkonu v prohlížeči Google Chrome pro vývojáře: 2, 8 s

**Velikost sestaveného JS souboru**

Velikost sestaveného JS souboru: 51,4 kB

**Komponenty**

Práce s komponentami je v projektech Vue klíčová, protože Vue je framework založený na komponentách, což umožňuje vývojářům strukturovat aplikaci jako soubor opakovaně použitelných komponent. Každá komponenta ve Vue projektech má svou vlastní strukturu, data a chování, což usnadňuje modularitu a opětovnou použitelnost kódu.

Vue CLI (příkazová řádka) je důležitá ve zjednodušení interakce s frameworkem a řízení různých aspektů projektu. Pomocí Vue CLI mohou vývojáři rychle generovat nové projekty, komponenty, služby a jiné části aplikace, což výrazně urychluje a zefektivňuje vývojový proces.

Níže je příklad hlavního souboru *App.js* z ukázkového projektu. Na příkladu lze vidět rozdělení komponent na tři části, a to část pro HTML šablonu, část pro styly a část pro javascriptový kód. Vše se nachází v jednom souboru, což přispívá k přehlednosti kódu a celkově projektu, kde se pracuje s více komponentami.

```vue
<template>
  <div
    class="flex flex-col min-h-screen font-Roboto bg-weather-primary"
  >
    <SiteNavigation />
    <RouterView class="flex-1" v-slot="{ Component }">
      <Transition name="page">
        <component :is="Component" />
      </Transition>
    </RouterView>
  </div>
</template>

<script setup>
import { RouterView } from "vue-router";
import SiteNavigation from "./components/SiteNavigation.vue";
</script>

<style>
.page-enter-active {
  transition: 600ms ease all;
}

.page-enter-from {
  opacity: 0;
}
</style>
```

*Obrázek 26 - Příklad komponenty ve Vue*

Bodové hodnocení: 9

### 4.4.2 Angular

Projekt s frameworkem Angular byl inicializován příkazem *ng new „název projektu“*. Po této inicializaci může uživatel zvolit, které funkcionality nebo moduly chce do projektu zahrnout, například Angular Router nebo SSR (Server-side Rendering) podporu. V případě tohoto projektu byly nainstalovány moduly jako Angular animations, Angular common, Angular compiler, Angular core, a další, což jsou základní stavební bloky pro každou Angular aplikaci. Pomocí příkazu npm install byly nainstalovány všechny potřebné závislosti pro projekt. V době psaní této práce je stabilní verzí Angular v17.1.0.

**Velikost frameworku**

Velikost frameworku v základní konfiguraci je 481 MB.

**Dokumentace**

Dokumentace Angularu nabízí zdroje informací, které pomáhají vývojářům všech úrovní zkušeností pochopit a efektivně využívat Angular. Dokumentace je rozdělena do přehledných sekcí, jako například Začínáme, kde je možné nalézt základní informace o tom, jak začít s Angular aplikacemi, včetně nastavení prostředí, prvních kroků s novým projektem a základního přehledu frameworku. Průvodce disponuje informacemi o základních konceptech, jako jsou komponenty a šablony a zároveň popisuje pokročilé techniky, jako injektování závislostí, routing a správu stavu. Nástroje obsahují popis Angular CLI, které pomáhá s automatizací mnoha úloh během vývojového procesu. Knihovny popisují jak využívat různé Angular knihovny, včetně podrobných návodů.

Bodové hodnocení: 7

**První vykreslení obsahu**

První vykreslení obsahu vypočítané pomocí nástroje Statistiky výkonu v prohlížeči Google Chrome pro vývojáře: 2,2 s

**Vykreslení největšího obsahu**

Vykreslení největšího obsahu vypočítáno pomocí nástroje Statistiky výkonu v prohlížeči Google Chrome pro vývojáře: 2,2 s

**Vykreslení obsahu – interaktivita**

Vykreslení obsahu – interaktivita vypočítáno pomocí nástroje Statistiky výkonu v prohlížeči Google Chrome pro vývojáře: 110 ms

**Index rychlosti**

Index rychlosti vypočítán pomocí nástroje Statistiky výkonu v prohlížeči Google Chrome pro vývojáře: 2,2 s

**Velikost sestaveného JS souboru**

Velikost sestaveného JS souboru: 277,8 kB

**Komponenty**

Práce s komponentami je klíčovým aspektem vývoje aplikací v Angularu, neboť Angular je framework, který staví na architektuře založené na komponentách. V Angularu se celá aplikace skládá z komponent, které jsou základními stavebními bloky pro uživatelská rozhraní. Každá komponenta v Angularu kombinuje šablonu HTML s třídou TypeScript, která obsahuje logiku komponenty, a případně i specifické styly. Zvláštností oproti frameworkům React a Vue je to, že při generování komponenty do projektu se vytvoří automaticky čtyři soubory od sebe oddělené, šablona HTML, typescriptový soubor, soubor pro stylizaci a soubor pro testování.

```
<div  class="flex-container">
<app-sitenavigation></app-sitenavigation>
<router-outlet></router-outlet>
</div>
```

*Obrázek 27 - Příklad šablony v hlavní komponentě*

```
import { Component } from '@angular/core';
import { RouterOutlet } from '@angular/router';
import { SitenavigationComponent } from './components/sitenavigation/sitenavigation.component';
import { HttpClientModule } from '@angular/common/http';


@Component({
    selector: 'app-root',
    standalone: true,
    templateUrl: './app.component.html',
    styleUrl: './app.component.scss',
    imports: [RouterOutlet, SitenavigationComponent, HttpClientModule ]
})
export class AppComponent {
  title = 'my-app';
}
```

*Obrázek 28 - Hlavní komponenta v Angular projektu*

Bodové hodnocení: 8

### 4.4.3 React

Projekt vytvořený s frameworkem React byl inicializován příkazem *npx create-react-app „název projektu“*. Po této inicializaci mohl uživatel přidávat různé knihovny nebo technologie podle potřeb projektu. V tomto konkrétním případě byla instalována knihovna React Router, která rozšiřuje základní funkčnost React aplikace. Pomocí příkazu *npm install* byly do projektu přidány závislosti jako axios pro práci s API a react-router-dom pro routování v aplikaci. V době psaní této práce je použitá stabilní verze Reactu v18.2.0 a React Bootstrapu ve verzi 2.10.0.

**Velikost frameworku**

Velikost frameworku v základní konfiguraci je 293 MB.

**Dokumentace**

Dokumentace Reactu je, stejně jako předchozí frameworky, do částí. Obsahuje sekci, která nabízí úvod do Reactu, vysvětluje základní koncepty a poskytuje průvodce pro instalaci a vytváření prvních React aplikací. Hlavní koncepty jsou popsány v samostatné kapitole, kde jsou vysvětleny základní stavební bloky React, jako jsou JSX, komponenty, props a stav. Dále poskytuje podrodné informace o API, včetně popisu komponent a hooků. React hookům je věnována samostatná kapitola, která vysvětluje, jak hooky používat pro správu stavů a dalších aspektů.

Bodové hodnocení: 8

**První vykreslení obsahu**

První vykreslení obsahu vypočítané pomocí nástroje Statistiky výkonu v prohlížeči Google Chrome pro vývojáře: 3 s

**Vykreslení největšího obsahu**

Vykreslení největšího obsahu vypočítáno pomocí nástroje Statistiky výkonu v prohlížeči Google Chrome pro vývojáře: 3,2 s

**Vykreslení obsahu – interaktivita**

Vykreslení obsahu – interaktivita vypočítáno pomocí nástroje Statistiky výkonu v prohlížeči Google Chrome pro vývojáře: 20 ms

**Index rychlosti**

Index rychlosti vypočítán pomocí nástroje Statistiky výkonu v prohlížeči Google Chrome pro vývojáře: 3,5 s

**Velikost sestaveného JS souboru**

Velikost sestaveného JS souboru: 210 kB

**Komponenty**

V Reactu je každá část uživatelského rozhraní reprezentována komponentou, která funguje jako samostatná jednotka s vlastní logikou a stavy. Každá komponenta v Reactu může obsahovat JSX, speciální syntaxi, která vypadá jako HTML a umožňuje psát značkovací kód přímo ve svých JavaScriptových souborech. Logika komponenty je typicky napsána v JavaScriptu a může být doplněna o lokální stav nebo reagovat na události, čímž se řídí interaktivita komponenty. Oproti Angularu, kde se komponenta skládá z několika souborů, v Reactu se obvykle celá komponenta nachází v jednom JavaScriptovém souboru, který zahrnuje jak logiku, tak i značkování JSX. Stylizace může být přiložena přímo v komponentě, nebo může být oddělena do externích CSS souborů. React rovněž podporuje koncept stavu (state) a vlastností (props), což umožňuje komponentám přijímat data z nadřazených komponent a reagovat na změny dat dynamicky.

```
import { Routes, Route, BrowserRouter } from "react-router-dom";
import SiteNavigation from "./components/SiteNavigation";
import "../src/custom.scss";
import HomeView from "./views/HomeView";
import AsyncCityView from "./components/AsyncCityView";

function App() {
  return (
    <div className="d-flex flex-column min-vh-100 fontanek bg-primary">
      <BrowserRouter>
        <SiteNavigation />
        <Routes>
          <Route path="/" element={<HomeView />} />
          <Route path="/AsyncCityView/:state/:city/:lat/:lng/:preview/:id" element={<AsyncCityView />} />
        </Routes>
      </BrowserRouter>
    </div>
  );
}

export default App;
```

*Obrázek 29 - Příklad hlavního souboru v Reactu App.js*

```
import { useState, Suspense } from "react";
import CityList from "../components/CityList";
import axios from "axios";
import { useNavigate } from "react-router-dom";
```

*Obrázek 30 - Příklad importu komponenty a hooků*

Bodové hodnocení: 8

## 4.5 Analýza kritérií (JS)

V této části práce se provádí analýza front-endových frameworků s využitím specificky vybraných kritérií. Pro přiřazení vah jednotlivým kritériím je využita metoda Saatyho, která

je vhodná v situaci, kdy hodnotící je pouze jedna osoba – autor práce. K vyhodnocení analýzy je aplikována metoda váženého součtu, o které literatura uvádí, že je přínosná pro hodnocení a umožňuje aplikaci na libovolný typ informací týkajících se preferenčních vztahů mezi prvky modelu (Šubrt, 2011).

### 4.5.1 Ohodnocení kritérií variant

V tabulce níže je prezentován podrobný přehled hodnocení kritérií pro různé varianty frontendových JS frameworků. Kritéria 2 a 8 jsou bodově ohodnocena a považuje se zde za ideální hodnotu ta nejvyšší. Zbylá kritéria, tedy kritéria číslo 1, 3, 4, 5, 6 a 7 jsou naměřená a za ideální hodnotu se považuje ta nejmenší.

*Tabulka 4 - Ohodnocení kritérií dle jednotlivých variant*

| **Framework** | | **Kritéria** | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|
| | | **1** | **2** | **3** | **4** | **5** | **6** | **7** | **8** |
| **1** | **Vue** | 107 | 10 | 2,8 | 3,4 | 0 | 2,8 | 51,4 | 9 |
| **2** | **Angular** | 481 | 7 | 2,2 | 2,2 | 110 | 2,2 | 277,8 | 8 |
| **3** | **React** | 293 | 8 | 3 | 3,2 | 20 | 3,5 | 210 | 8 |
| **Povaha** | | min | max | min | min | min | min | min | max |

### 4.5.2 Určení vah kritérií (Saatyho metoda)

Podle metody Saatyho byly stanoveny a vypočítány váhy pro jednotlivá kritéria. Každé kritérium je porovnáno s ostatními, což umožňuje určit jeho relativní důležitost a vliv. Pro další analýzy je klíčový výpočet normalizovaných hodnot váh kritérií, přičemž suma těchto normalizovaných vah by měla být rovna jedné, což slouží jako kontrolní mechanismus. Celkový souhrn je v tabulce č. 5.

*Tabulka 5 - Určené váhy kritérií dle Saatyho metody*

| | | | | | | | | | Geometrický průměr ($b_i$) | Váha ($v_i$) |
|---|---|---|---|---|---|---|---|---|---|---|
| **Kritéria** | **1** | **2** | **3** | **4** | **5** | **6** | **7** | **8** | | |
| **1** | 1 | 1/3 | 1/3 | 3 | 1/5 | 1/9 | 1/7 | 1/5 | 0,347 | 0,028 |
| **2** | 3 | 1 | 1/3 | 3 | 1/3 | 1/5 | 1/5 | 1/3 | 0,583 | 0,047 |
| **3** | 5 | 3 | 1 | 3 | 1 | 1/7 | 1/7 | 1/3 | 0,862 | 0,069 |
| **4** | 1/3 | 1/3 | 1/3 | 1 | 1/5 | 1/9 | 1/7 | 1/7 | 0,253 | 0,020 |
| **5** | 5 | 3 | 1 | 5 | 1 | 1/5 | 1/5 | 1/3 | 1 | 0,080 |
| **6** | 9 | 5 | 7 | 9 | 5 | 1 | 1 | 5 | 4,039 | 0,324 |
| **7** | 7 | 5 | 7 | 7 | 5 | 1 | 1 | 5 | 3,793 | 0,305 |
| **8** | 5 | 3 | 3 | 7 | 3 | 1/5 | 1/5 | 1 | 1,575 | 0,126 |
| | | | | | | | | **Suma** | **12,453** | **1** |

Pomocí výpočtů získaných z jednotlivých vah kritérií je možné ve výsledné tabulce vidět nejvyšší preference. Nejvyšší preferenci má kritérium číslo 6, tedy index rychlosti. Důležitost výkonu při vytváření webových aplikací byla popsána v kapitole 4.5. Jednotka index rychlosti výborným způsobem demonstruje výkonnostní stránku webových stránek a aplikací. V zásadě se jedná o to, jak rychle se viditelně vyplňuje obsah stránky. Druhým kritériem s nejvyšší preferencí je kritérium číslo 7, tedy velikost sestaveného JS souboru. Velikost javascriptového souboru úzce souvisí s výkonem webových aplikací. Platí, že čím velikostně menší soubory musí stránka stáhnout, tím rychleji je načte a zobrazí obsah uživateli. Pokud stránka nebo webová aplikace nereaguje promptně na vstupy uživatele, uživatel je velice snadno a rychle znepokojen a opustí stránku. Naopak nejnižší preferenci má kritérium číslo 4, tedy vykreslení největšího obsahu.

### 4.5.3 Analýza metodou váženého součtu

Při provádění analýzy metodou váženého součtu je klíčové určit pro každé kritérium ideální (*H*) a bazální (*D*) varianty. Tento výběr je přímo propojen s tím, zda je cílem kritéria minimalizace či maximalizace hodnot. Například pro kritérium číslo 1, které se týká velikosti frameworku a je minimalizační, je jako optimální hodnota zvolena ta nejnižší ze všech možností a jako bazální pak ta nejvyšší. U kritérií, které jsou maximalizační, je situace

obrácená, kde ideální variantou je ta s nejvyšší hodnotou a bazální s tou nejnižší. Ideální a bazální varianty pro jednotlivá kritéria jsou specifikovány v další tabulce.

*Tabulka 6 - Stanovení ideální a bazální varianty*

| | **Kritérium** | | | | | | | |
|---|---|---|---|---|---|---|---|---|
| | **1** | **2** | **3** | **4** | **5** | **6** | **7** | **8** |
| **Ideální varianta** | 107 | 10 | 2,2 | 2.2 | 0 | 2,2 | 51,4 | 9 |
| **Bazální varianta** | 481 | 7 | 3 | 3,4 | 110 | 3,5 | 277,8 | 8 |

Pro posouzení jednotlivých variant podle stanovených kritérií se konstruuje kriteriální matice R. Vycházející z matematického vzorce popsaného v sekci 3.5.2 této práce, postupně dochází k výpočtu částečných užitků pro každou variantu a kritérium. Tento proces pokračuje až do sestavení kompletní kriteriální matice R. Například pro variantu 2 a kritérium 1 se vypočítá dosazením do vzorce následovně:

$$r_{2.1} = \frac{481 - 481}{107 - 481} = 0$$

*Tabulka 7 - Standardizovaná kriteriální matice*

| | **Kritérium** | | | | | | | |
|---|---|---|---|---|---|---|---|---|
| | **1** | **2** | **3** | **4** | **5** | **6** | **7** | **8** |
| **Vue** | 1,000 | 1,000 | 0,250 | 0,000 | 1,000 | 0,538 | 1,000 | 1,000 |
| **Angular** | 0,000 | 0,000 | 1,000 | 1,000 | 0,000 | 1,000 | 0,000 | 0,000 |
| **React** | 0,503 | 0,333 | 0,000 | 0,167 | 0,818 | 0,000 | 0,300 | 0,000 |

Abychom získali celkový užitek pro danou variantu, je nezbytné sečíst hodnoty příslušného řádku (který představuje konkrétní variantu) matice. Při tomto součtu je třeba každou hodnotu z řádku násobit její normalizovanou vahou kritéria. Následuje příklad, kde je ukázán výpočet celkového užitku pro vybranou variantu.

*Celkový užitek varianty 1:*

1 * 0,028 + 1 * 0,047 + 0,25 * 0,069 + 0 * 0,02 + 1 * 0,08 + 0,538 * 0,325 + 1 * 0,305 + 1 * 0,126 = **0,778**

*Tabulka 8 - Celkový užitek variant a jejich pořadí*

| | **Celkový užitek variant** | **Pořadí** |
|---|---|---|
| **Vue** | 0,778 | 1. |
| **Angular** | 0,414 | 2. |
| **React** | 0,190 | 3. |

**Výsledné vyhodnocení**

Byl proveden rozbor a porovnání různých JavaScriptových frameworků s cílem najít nejvhodnější framework pro vývoj aplikace. Celkem bylo posuzováno pět frameworků: Angular, React, Vue, Ember a Svelte. Výběr nejlepšího frameworku se uskutečnil ve dvou krocích. V prvním kroku byly z kandidátů odstraněny ty frameworky, které podle údajů ze StackOverflow a šetření prováděným Sacha Greif a Eric Burel, nevykazují významnou podporu od vývojářských komunit nebo nejsou v České republice dostatečně rozšířené mezi freelancery a jejich dovednosti nejsou často vyžadovány na portálu Jobs.cz. Z tohoto důvodu byly z výběru vyřazeny frameworky Ember a Svelte. Ve druhé fázi zůstaly ve výběru Angular, React a Vue, u kterých bylo provedeno podrobnější srovnání z hlediska funkcí a výkonností na konkrétním projektu, jež nabízejí vývojářům, a to pomocí vícekriteriální analýzy variant. Tato analýza s využitím metody váženého součtu určila Vue jako nejlepší možnost s hodnocením celkové užitečnosti 0,778. Na základě této analýzy byl framework Vue zvolen jako preferovaný framework pro vývoj aplikace.

## 4.6 Charakteristika hodnotících kritérií (UI)

UI frameworky jsou nástroje založené primárně na CSS, které přinášejí významné výhody v oblasti tvorby responzivních designů pro webové aplikace. V této studii jsou analyzovány a porovnávány různé frameworky, mezi něž patří Bootstrap, Tailwind CSS, Foundation, Bulma a Skeleton. Bootstrap a Foundation představují rozsáhlé sady nástrojů s bohatou nabídkou CSS tříd, které usnadňují vývoj uživatelských rozhraní pro webové aplikace. Tyto dva frameworky nabízejí podobné funkce a často jsou vnímány jako vzájemní konkurenti, což znamená, že výběr mezi nimi může být z velké části záležitostí osobní preference. Na

druhé straně Skeleton představuje "lehčí" řešení, což znamená, že poskytuje méně funkcí, ale zároveň je jeho velikost menší, což může být výhodou pro rychlejší načítání stránek. K určení nejvhodnějšího UI frameworku byla použita vícekriteriální analýza, která zahrnovala kritéria uvedená v následující tabulce.

*Tabulka 9 - Kritéria pro vícekriteriální analýzu UI frameworků*

| Číslo | Kritérium | Jednotka |
|---|---|---|
| **1** | Podpora webových prohlížečů | Bodové hodnocení |
| **2** | Mřížka a zlomové body | |
| **3** | Responzivní prvky | |
| **4** | Customizace | |
| **5** | Dokumentace | |
| **6** | Velikost | kB |

Kritérium číslo 5, týkající se velikosti frameworku, je kvantitativní s cílem minimalizace. Informace pro toto kritérium se shromažďují stahováním minifikovaných verzí CS. Na druhé straně, zbývající kritéria jsou kvalitativní a směřují k maximalizaci hodnot. Hodnocení těchto kritérií se odvíjí významně od subjektivních postřehů, protože při porovnávání konkurenčních frameworků není možné stanovit absolutní objektivní měřítko pro určení, který framework je lepší nebo horší. Pro bodové hodnocení těchto kvalitativních kritérií se používá škála od 1 do 10, přičemž 1 značí nejnižší a 10 nejvyšší hodnocení.

**Podpora webových prohlížečů**

Důležitost podpory webových prohlížečů u UI frameworků nesmí být podceňována, jelikož může zásadně ovlivnit rozhodnutí uživatelů využívat danou aplikaci. Je klíčové zvážit, zda daný framework poskytuje podporu pro novější verze prohlížečů, starší verze, nebo obojí. Zvláštní pozornost by měla být věnována tomu, jak se starší prohlížeče vyrovnávají s moderními funkcemi daného frameworku, jako je například responzivní design.

**Mřížka a zlomové body**

Mřížkový systém a zlomové body jsou klíčové prvky každého UI frameworku umožňujícího tvorbu responzivních designů. Mřížkový systém (Grid System) je navržen tak, aby umožňoval umisťování různých prvků do buněk na webové stránce a adaptaci jejich velikosti (typicky šířky) podle specifik zařízení, na kterém je web zobrazován. Zlomové body (Breakpoints) jsou následně využívány pro modifikaci designu prvků v závislosti na šířce

obrazovky, což se realizuje s pomocí mediálních dotazů (Media Queries). Efektivitu práce se zlomovými body může značně zvýšit použití CSS preprocesoru, protože umožňuje jednoduchou definici nebo využití standardních zlomových bodů v rámci preprocesoru.

**Responzivní prvky**

Responzivní ovládací prvky jsou navrženy tak, aby poskytovaly komplexní funkčnost na různých zařízeních, což umožňuje webu přizpůsobit se různým velikostem a typům obrazovek. Typickým příkladem může být navigační lišta, která na stolních počítačích a často i tabletech bývá zobrazena horizontálně v hlavičce stránky a je stále viditelná, na rozdíl od mobilních zařízení, kde se navigace obvykle skrývá a zobrazuje vertikálně po interakci s tlačítkem. Toto kritérium se zaměřuje na posouzení, zda různé frameworky ve svém standardním nastavení poskytují takovéto responzivní ovládací prvky.

**Customizace**

Toto kritérium hodnotí, zda a jak lze implementovat rozsáhlé změny CSS stylů napříč celým webem. Přínosné je, když podporovaný framework nabízí integraci s alespoň jedním CSS preprocesorem. Dále se zkoumá možnost individuálního nastavení stylů frameworku před jeho stahováním, což umožňuje uživatelům modifikovat a stahovat verzi frameworku přizpůsobenou jejich specifickým potřebám.

**Dokumentace**

Vzhledem k rozmanitosti a aplikacím frameworků je klíčové prozkoumat poskytovanou dokumentaci od vývojářů nástroje. Je žádoucí, aby dokumentace obsahovala kompletní informace a pokyny pro používání komponent. Při bodovém hodnocení bude rozhodující především úroveň, uspořádání a komplexnost poskytnuté dokumentace.

**Velikost**

Vzhledem k tomu, že knihovny, které se integrují do projektu, zahrnují rozsáhlý zdrojový kód, velikost frameworku může nepřímo reflektovat jeho schopnosti, možnosti úprav a funkčnost. Velké objemy frameworkových souborů mohou významně ovlivnit celkovou velikost webu nebo aplikace, což může vést ke zpomalení načítání. Vzhledem k zaměření knihoven na responzivní design je důležité brát v úvahu i používání webu na mobilních zařízeních, která se často připojují k internetu přes mobilní data s omezeným datovým limitem. Proto je vhodné minimalizovat velikost dat, které návštěvníci musí stahovat, což zahrnuje používání pouze minifikovaných verzí importovaných souborů a optimalizaci celkové velikosti frameworku, včetně všech doporučených knihoven. Velikosti jsou výsledky stažených minifikovaných souborů z CDN (Held, 2020).

### 4.6.1 Bootstrap

Bootstrap je framework pro design webových uživatelských rozhraní, který byl původně vyvinut v roce 2011 zaměstnanci společnosti Twitter, Markem Ottou a Jacobem Thorntonem. V současnosti je Bootstrap nabízen jako open-source řešení a řadí se mezi nejpoužívanější nástroje v oblasti webového vývoje (Jakobus, 2018, s. 18). V době psaní této práce byla aktuální verzi Bootstrapu verze 5, konkrétněji 5.3.3. Dokumentace Bootstrapu je dostupná na getbootstrap.com/docs.

**Podpora webových prohlížečů**

Bootstrap poskytuje podporu pro většinu webových prohlížečů používaných dnes širokou škálou uživatelů, a to jak pro mobilní zařízení, tak pro desktopové počítače. Informace o tom, které prohlížeče jsou kompatibilní s Bootstrapem, lze nalézt v následujících tabulkách, které toto zobrazují přehledně.

| | Chrome | Firefox | Safari | Android Browser & WebView |
|---|---|---|---|---|
| **Android** | Supported | Supported | — | v6.0+ |
| **iOS** | Supported | Supported | Supported | — |

*Obrázek 31 - Podpora Bootstrapu na mobilních zařízeních (Bootstrap, 2024)*

| | Chrome | Firefox | Microsoft Edge | Opera | Safari |
|---|---|---|---|---|---|
| **Mac** | Supported | Supported | Supported | Supported | Supported |
| **Windows** | Supported | Supported | Supported | Supported | — |

*Obrázek 32 - Podpora Bootstrapu na desktopech (Bootstrap, 2024)*

Bodové hodnocení: 9

**Mřížka a zlomové body**

Mřížkový systém Bootstrapu umožňuje snadno vytvářet rozložení stránek, které jsou automaticky přizpůsobené různým velikostem obrazovek zařízení. Mřížka je založena na řadách a sloupcích, kde sloupce definují šířku obsahu a řady uspořádání horizontálního layoutu.

Zlomové body v Bootstrapu definují rozlišení, při kterých se design stránky mění, aby byl lépe čitelný a uživatelsky přívětivější na různých zařízeních. Tyto zlomové body umožňují určit, jak se stránka nebo její prvky budou zobrazovat na mobilních telefonech, tabletech, laptopech a dalších zařízeních. Bootstrap standardně zahrnuje několik zlomových bodů pro

různé šířky obrazovky, což usnadňuje implementaci responzivního designu bez potřeby složitého kódování. Následující tabulka znázorňuje přesné hodnoty zlomových bodů a jejich třídu, které je možné při vývoji použít.

| Breakpoint | Class infix | Dimensions |
|---|---|---|
| Extra small | *None* | <576px |
| Small | sm | ≥576px |
| Medium | md | ≥768px |
| Large | lg | ≥992px |
| Extra large | xl | ≥1200px |
| Extra extra large | xxl | ≥1400px |

*Obrázek 33 - Zlomové body v Bootstrapu (Bootstrap, 2024)*

Bodové hodnocení: 9

**Responzivní prvky**

Responzivní prvky v Bootstrapu se automaticky přizpůsobují velikosti a rozlišení obrazovky, na které jsou zobrazeny, což zajišťuje, že webová aplikace nebo stránka bude fungovat efektivně a bude snadno použitelná jak na velkých monitorových obrazovkách, tak na malých mobilních zařízeních. Bootstrap obsahuje širokou škálu responzivních ovládacích prvků, včetně tlačítek, formulářových prvků, navigačních lišt, dropdown menu a dalších komponent, které se dají snadno implementovat a přizpůsobit. Díky integrovaným CSS třídám a responzivním utilitám je možné rychle vytvářet rozhraní, které reaguje na změny v rozlišení obrazovky a orientaci zařízení.

Bodové hodnocení: 8

**Customizace**

Customizace v Bootstrapu umožňuje přizpůsobit vzhled a chování webových stránek, aby odpovídaly specifickým požadavkům. Díky flexibilitě Bootstrapu je možné upravovat širokou škálu aspektů, od barev, písem a tlačítek, až po složitější komponenty a layouty. Customizace může probíhat na různých úrovních, od jednoduchých změn pomocí přetížení vestavěných CSS tříd přes použití SASS proměnných pro generování nových stylů až po upravování zdrojových souborů Bootstrapu pro komplexní změny. Bootstrap také nabízí nástroj na svém webu, kde je možné vybrat pouze ty komponenty, které jsou potřeba, a přizpůsobit základní barevné schéma a typografii před stažením frameworku.

Bodové hodnocení: 8

**Dokumentace**

Dokumentace Bootstrapu poskytuje detailní a jasně strukturované informace o tom, jak využít tento framework k maximálnímu potenciálu. Nejen že zahrnuje instrukce k instalaci a základnímu používání, ale také nabízí podrobné průvodce pro práci s různými komponenty, utilitami a jQuery pluginy. V dokumentaci Bootstrapu lze nalézt konkrétní příklady kódu, které usnadňují pochopení, jak komponenty fungují a jak je lze implementovat do vlastních projektů. Navíc jsou zde uvedeny i pokyny pro přizpůsobení a rozšíření stávajících stylů, což umožňuje vývojářům upravit vzhled a chování prvků podle specifických potřeb jejich webové stránky nebo aplikace.

Bodové hodnocení: 9

**Velikost**

Velikost Bootstrapu v základní minifikované konfiguraci je 164 kB.

### 4.6.2 Tailwind CSS

Tailwind CSS nabízí rychlý a snadný způsob, jak aplikovat styly na webové stránky. Jedná se o framework, který dává přednost utilitám a umožňuje expresní tvorbu vlastních uživatelských rozhraní. Tailwind je velmi flexibilní a nabízí základní stavební prvky pro tvorbu designu bez nutnosti zásahů do předdefinovaných stylů (Alif, 2023). V době psaní této práce je aktuální verze Tailwindu 3.4.1.

**Podpora webových prohlížečů**

Obecně platí, že Tailwind CSS verze 3.0 je navržen a testován pro nejnovější stabilní verze prohlížečů Chrome, Firefox, Edge a Safari. Nepodporuje žádnou verzi prohlížeče Internet Explorer, včetně IE 11 (Tailwindcss, 2024).

Bodové hodnocení: 8

**Mřížka a zlomové body**

V Tailwindu je mřížkový systém založen na flexboxu a CSS gridu, což umožňuje vytvářet komplexní rozložení stránek s přesnou kontrolou nad chováním každého prvku v různých velikostech obrazovek. V základu se mřížkový systém dělí v Tailwindu na 12 sloupců.

Zlomové body v Tailwind CSS jsou předem definovány a lze je snadno upravit nebo rozšířit podle potřeb projektu. Tyto zlomové body umožňují aplikovat různé styly v závislosti na velikosti obrazovky, což je klíčové pro tvorbu responzivních webových aplikací, které

efektivně fungují na široké škále zařízení od mobilních telefonů po velké desktopové monitory. Níže jsou popsány zlomové body z oficiální dokumentace TailwindCSS (Tailwindcss, 2024).

| Breakpoint prefix | Minimum width | CSS |
|---|---|---|
| `sm` | 640px | `@media (min-width: 640px) { ... }` |
| `md` | 768px | `@media (min-width: 768px) { ... }` |
| `lg` | 1024px | `@media (min-width: 1024px) { ... }` |
| `xl` | 1280px | `@media (min-width: 1280px) { ... }` |
| `2xl` | 1536px | `@media (min-width: 1536px) { ... }` |

*Obrázek 34 - Zlomové body v Tailwindu (Tailwindcss, 2024)*

Bodové hodnocení: 9

**Responzivní prvky**

Ve výchozím nastavení používá Tailwind systém zlomových bodů zaměřený na mobilní zařízení jako první, což je podobné tomu, na co byste mohli být zvyklí v jiných frameworkách, jako je například Bootstrap (Tailwindcss, 2024).

Responzivní prvky Tailwindu umožňují rychle vytvářet webové stránky a aplikace, které se bezproblémově přizpůsobí jakémukoli rozlišení či velikosti zařízení. Díky systému založenému na utilitních třídách je možné jednoduše aplikovat různé styly na elementy závisle na breakpointech, což přináší kontrolu nad tím, jak se obsah zobrazí na různých zařízeních.

Bodové hodnocení: 9

**Customizace**

S Tailwindem je možné upravit téměř každý aspekt designu, od barev, písem až po stíny a interakce, vše prostřednictvím konfiguračního souboru. Tento přístup znamená, že namísto přepisování stávajících stylů je možné vytvořit vlastní designový systém od základu, který bude odpovídat specifickým požadavkům.

Bodové hodnocení: 10

**Dokumentace**

V dokumentaci TailwindCSS se nachází komplexní průvodce, který pokrývá vše od základní instalace, konfigurace, přes použití různých utilit a komponent, až po pokročilé techniky pro práci s customizací. Každý aspekt frameworku je podrobně vysvětlen s příklady kódu, které

ilustrují praktické použití v reálných scénářích. Kromě toho dokumentace poskytuje zdroje pro učení a inspiraci, včetně odkazů na projekty, ukázkové šablony a časté otázky, což novým uživatelům usnadňuje začlenění TailwindCSS do jejich projektů a zkušeným vývojářům poskytuje materiál pro další rozvoj jejich dovedností.

Bodové hodnocení: 9

**Velikost**

Velikost Tailwind v základní minifikované konfiguraci je 509 kB.

### 4.6.3 Foundation

Od roku 2019 se framework Foundation, jenž byl dříve ve správě společnosti ZURB, stal open-source projektem, na jehož údržbě se podílejí dobrovolníci. Tento framework upřednostňuje přístup "mobile-first" a je zvláště vhodný pro vytváření rozsáhlých webových aplikací vyžadujících unikátní design (Bose, 2011). V době psaní práce je aktuální verzí verze 6.8.1.

**Podpora webových prohlížečů**

Framework Foundation prochází testováním v širokém spektru prohlížečů a zařízení a je kompatibilní s verzemi až po Internet Explorer 9 a Android 2 (Foundation, 2020). Níže je tabulka s přehledem o podpoře frameworku jednotlivými webovými prohlížeči.

| | |
|---|---|
| Chrome | ✓ Last Two Versions |
| Firefox | |
| Safari | |
| Opera | |
| Mobile Safari[1] | |
| IE Mobile | |
| Edge | |
| Internet Explorer | ✓ Versions 9+ |
| Android Browser | ✓ Versions 4.4+ |

*Obrázek 35 - Podpora Foundation webovými prohlížeči (Foundation, 2020)*

Bodové hodnocení: 9

**Mřížka a zlomové body**

Foundation Framework zahrnuje sadu předdefinovaných zlomových bodů, které jsou vhodné pro řadu standardních zařízení. Vývojáři mají také možnost přidávat vlastní zlomové

body, což umožňuje ještě větší flexibilitu a kontrolu nad responzivním chováním webových stránek. Ovšem předdefinované zlomové body jsou pouze tři. A to (Foundation, 2020):

- Small: jakákoliv velikost obrazovky
- Medium: jakákoliv velikost obrazovky vetší nebo rovna 640 px a menší než 1024 px
- Large: jakákoliv velikost obrazovky větší nebo rovna 1024 px

Jako Bootstrap disponuje v základní konfiguraci mřížkového systému 12 sloupci.

Bodové hodnocení: 6

**Responzivní prvky**

Foundation nabízí širokou škálu responzivních komponent, včetně navigačních lišt, formulářových prvků, tlačítek, tabulek a dalších, které automaticky mění svou velikost a layout v závislosti na velikosti obrazovky zařízení. Tento přístup zajišťuje, že webové stránky vytvořené s Foundationem poskytují konzistentní uživatelský zážitek napříč všemi platformami.

Bodové hodnocení: 7

**Customizace**

Díky použití SASS preprocesoru je možné upravit styly, barvy, mřížky a další komponenty. Foundation také umožňuje detailní úpravy pomocí systému nastavení a mixinů, což umožňuje flexibilitu a kontrolu nad designem, zatímco zachovává robustní a responzivní základ.

Customizace před samotným stažením frameworku prostřednictvím interaktivního konfiguračního nástroje není u Foundation Frameworku možná. To představuje omezení ve srovnání s Bootstrapem, který takovou možnost nabízí a umožňuje předem upravit nastavení pomocí interaktivního formuláře.

Bodové hodnocení: 7

**Dokumentace**

Dokumentace Foundation se zaměřuje na flexibilitu a modulárnost. Na rozdíl od některých jiných frameworků, které mohou být více přednastavené, Foundation poskytuje možnosti pro customizaci a přizpůsobení, což je v dokumentaci důkladně popsáno. Dále je zde kladen důraz na responzivní design a přístupnost. Obecně je dokumentace Foundation zaměřena na flexibilní řešení.

Bodové hodnocení: 8

**Velikost**

Velikost Foundation v základní minifikované konfiguraci je 133 kB.

### 4.6.4 Bulma

V době psaní práce je aktuální verzí frameworku Bulma 0.9.4. Jedná se framework s přístupem mobile first.

**Podpora webových prohlížečů**

Bulma je navržena tak, aby byla kompatibilní s novými verzemi prohlížečů jako Chrome, Edge, Firefox, Opera a Safari. Na druhou stranu, podpora pro starší prohlížeče, konkrétně pro Internet Explorer od verze 10 výše, je omezená. K dosažení kompatibility s funkcemi Flexboxu ve starších verzích prohlížečů Bulma využívá nástroj Autoprefixer (Tripathi, 2024).

Bodové hodnocení: 7

**Mřížka a zlomové body**

V základu disponuje Bulma 12 sloupci. Bulma nabízí v základní konfiguraci 4 zlomové body, těmi jsou (BULMA, 2024):

- Mobile: do velikosti 768 px
- Tablet: od velikosti 759 px
- Desktop: od velikosti 1024 px
- Widescreen: od velikosti 1216 px
- Fullhd: od velikosti 1408 px

Bodové hodnocení: 7

**Responzivní prvky**

Bulma také nabízí poměrně hodně responzivních prvků, jako například komponenty Card, Modal, Message, Navbar, Menu, Dropdown (BULMA, 2024).

Bodové hodnocení: 6

**Customizace**

Díky 419 Sass proměnným rozděleným do 28 souborů nabízí Bulma flexibilitu a kontrolu nad stylem webových stránek a aplikací. Je možné snadno upravit barvy, fonty, rozměry a další aspekty (BULMA, 2024).

Bulma klade důraz na minimalismus. To znamená, že ve výchozím nastavení mám méně přednastavených komponent, což poskytuje větší prostor pro customizaci.

Bodové hodnocení: 7

**Dokumentace**

Minimalističtější přístup Bulmy se odráží i v dokumentaci. Kategorií a sekcí zde není tolik jako u předešlých frameworků. Je zde celkový přehled, customizace, utility, sloupce, elementy, komponenty, formuláře, layout a to jsou všechny kategorie, které se ve stromové struktuře dokumentace nacházejí.

Bodové hodnocení: 7

**Velikost**

Velikost frameworku Bulma je v základní minifikované konfiguraci 207 kB.

### 4.6.5 Skeleton

Bose (2011) charakterizuje Skeleton ne jako obvyklý CSS framework, ale spíše jako "velmi jednoduchou responzivní šablonu" pro webové designy. Skeleton se vyznačuje svým minimalistickým designem, což je patrné z jeho velikosti – obsahuje pouze 400 řádků kódu. Aktuální verzí, při psaní této práce je 2.0.4.

**Podpora webových prohlížečů**

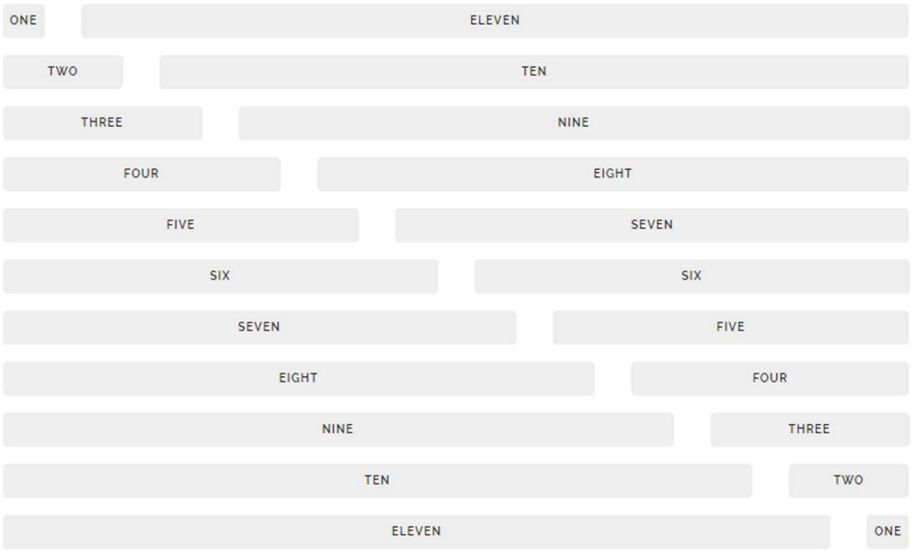
Framework Skeleton je designován jako lehký CSS framework, jehož hlavním cílem je poskytnout základní stylování a responzivní layout pro rychlý start webových projektů. Jeho podpora prohlížečů je všeobecně dobrá, zejména ve většině moderních prohlížečů, včetně posledních verzí Chrome, Firefox, Safari a Edge. Nicméně podpora pro starší prohlížeče, jako je Internet Explorer může být omezena nebo nekompletní.

Bodové hodnocení: 9

**Mřížka a zlomové body**

Skeleton poskytuje základní mřížkový systém s 12 sloupci. Na rozdíl od frameworků jako Bootstrap nebo Foundation, Skeleton neumožňuje jednoduché přidávání nebo odebírání

sloupců. Níže je obrázek z oficiální dokumentace, který demonstruje 12 sloupců mřížkového systému.




*Obrázek 36 - Mřížka Skeleton (Skeleton, 2024)*

Zlomových bodů má k dispozici Skeleton 5. Následující tabulka znázorňuje zlomové body.

- **Desktop HD**: 1200px
- **Desktop**: 1000px
- **Tablet**: 750px
- **Phablet**: 550px
- **Mobile**: 400px

*Obrázek 37 - Zlomové body ve Skeletonu (Skeleton, 2024)*

Bodové hodnocení: 7

**Responzivní prvky**

Ve standardní sadě Skeleton se nachází pár responzivních prvků, jako například tlačítko, formulář, seznam a tabulky.

Bodové hodnocení: 3

**Customizace**

Skeleton je navržen jako minimalistický nástroj pro rychlý vývoj webových rozhraní, jehož primární zaměření je na mřížkový systém a zlomové body. Customizace v rámci Skeletonu je tedy možná, avšak převážně se omezuje na úpravy těchto základních prvků. Může být upravena šířku sloupců, mezery mezi nimi, reakci mřížky na různé zlomové body a další aspekty, které ovlivňují vizuální strukturu webové stránky.

Bodové hodnocení: 3

**Dokumentace**

Dokumentace Skeleton je velmi stručn, což odráží minimalistickou povahu samotného frameworku. Nabízí základní přehled o tom, jak používat mřížkový systém a jak implementovat zlomové body, které jsou hlavními stavebními bloky tohoto nástroje. Je zde možné najít jasné příklady a jednoduché instrukce, které umožní rychle se s frameworkem seznámit a začít ho efektivně využívat ve svých projektech. Zároveň je zde možné nalézt jeden příklad využití Skeletonu.

Bodové hodnocení: 4

**Velikost**

Velikost frameworku Skeleton je ve verzi 2.0.4 5,8 kB.

## 4.7 Analýza kritérií (UI)

Stejně jako u javascriptových frameworků v kapitole 4.5, i zde se budou UI frameworky porovnávat na základě specificky zvolených kritérií.

### 4.7.1 Ohodnocení kritérií variant

V následující tabulce je uvedeno podrobné hodnocení kritérií pro různé možnosti frontendových UI frameworků. Pro kritérium číslo 6 (velikost) je nejnižší hodnota považována za ideální. Ostatní kritéria, specificky čísla 1, 2, 3, 4 a 5 jsou bodově ohodnocena a zde se za ideální hodnotu považuje ta nejvyšší.

*Tabulka 10 - Ohodnocení kritérií dle jednotlivých variant*

| **Framework** | | **Kritéria** | | | | | |
|---|---|---|---|---|---|---|---|
| | | **1** | **2** | **3** | **4** | **5** | **6** |
| **1** | **Bootstrap** | 9 | 9 | 8 | 8 | 9 | 164 |
| **2** | **Tailwind** | 8 | 9 | 9 | 10 | 9 | 509 |
| **3** | **Foundation** | 9 | 6 | 7 | 7 | 8 | 133 |
| **4** | **Bulma** | 7 | 7 | 6 | 7 | 7 | 207 |
| **5** | **Skeleton** | 9 | 7 | 3 | 3 | 4 | 6 |
| **Povaha** | | max | max | max | max | max | min |

### 4.7.2 Určení vah kritérií (Saatyho metoda)

V metodě Saatyho jsou přiřazeny a spočítány váhy pro každé z kritérií. Díky vzájemnému porovnání kritérií lze odhadnout jejich relativní význam a vliv na celkové hodnocení. Zásadním krokem v analýze je výpočet normalizovaných hodnot pro tyto váhy, přičemž

celkový součet normalizovaných vah by měl dosahovat hodnoty jedna. Detaily tohoto procesu jsou prezentovány v tabulce níže.

*Tabulka 11 - Určené váhy kritérií dle Saatyho metody*

| | | | | | | | **Geometrický průměr ($b_i$)** | **Váha ($v_i$)** |
|---|---|---|---|---|---|---|---|---|
| **Kritéria** | **1** | **2** | **3** | **4** | **5** | **6** | | |
| **1** | 1 | 3 | 1/7 | 1/5 | 3 | 1/3 | 0,66 | 0,07 |
| **2** | 1/3 | 1 | 1/9 | 1/7 | 1 | 1/5 | 0,32 | 0,04 |
| **3** | 7 | 9 | 1 | 3 | 7 | 3 | 3,98 | 0,44 |
| **4** | 5 | 7 | 1/3 | 1 | 5 | 3 | 2,37 | 0,26 |
| **5** | 1/3 | 1 | 1/7 | 1/5 | 1 | 1/3 | 0,38 | 0,04 |
| **6** | 3 | 5 | 1/3 | 1/3 | 3 | 1 | 1,31 | 0,14 |
| | | | | | | **Suma** | **9,02** | **1** |

Váhy jednotlivých kritérií jsou založeny především na poznatcích získaných studiem odborných zdrojů. Bose (2011) uvádí jako nejdůležitější kritérium při porovnávání UI frameworků responzivní prvky a obecně nástroje, kterými jednotlivé UI frameworky disponují. Dále uvádí customizaci mezi pěti důležitými kritérii. Výkon a podporu webových prohlížečů zařadil také mezi pět nejdůležitějších kritérií, kterými se řídit při výběru UI frameworku. Dokumentaci naopak nejmenuje vůbec, z toho důvodu je tomuto kritériu přisuzována nejmenší váha.

### 4.7.3 Analýza metodou váženého součtu

Při použití metody váženého součtu, stejně jako v kapitole 4.5.3 je nezbytné stanovit pro každé kritérium jeho ideální (*H*) a bazální (*D*) hodnoty, což závisí na tom, zda má být dané kritérium minimalizováno nebo maximalizováno. Jako příklad, u minimalizačního kritéria číslo 6, které hodnotí velikost frameworku, je nejlepší (ideální) hodnota ta nejmenší a bazální hodnota je naopak největší z dané sady. U maximalizačních kritérií je tento přístup inverzní: nejlepší hodnota je ta nejvyšší a bazální je nejnižší. Konkrétní ideální a bazální hodnoty pro všechna kritéria jsou uvedeny v následující tabulce.

*Tabulka 12 - Stanovení ideální a bazální varianty*

| | Kritérium | | | | | |
|---|---|---|---|---|---|---|
| | **1** | **2** | **3** | **4** | **5** | **6** |
| **Ideální varianta** | 9 | 9 | 9 | 10 | 9 | 6 |
| **Bazální varianta** | 7 | 6 | 3 | 3 | 4 | 509 |

Pro posouzení jednotlivých variant podle stanovených kritérií se konstruuje kriteriální matice R. Vycházející z matematického vzorce popsaného v sekci 3.5.2 této práce, postupně dochází k výpočtu částečných užitků pro každou variantu a kritérium. Tento proces pokračuje až do sestavení kompletní kriteriální matice R. Například pro variantu 2 a kritérium 1 se vypočítá dosazením do vzorce následovně:

$$r_{2.1} = \frac{8 - 7}{9 - 7} = 0{,}5$$

Hodnota 8 je hodnota $y_{ij}$, tedy v tomto případě hodnota prvního kritéria u druhé varianty. Hodnota 7 představuje bazální variantu $d_j$ a hodnota 9 variantu ideální $h_j$.

*Tabulka 13 - Standardizovaná kriteriální matice*

| | Kritérium | | | | | |
|---|---|---|---|---|---|---|
| | **1** | **2** | **3** | **4** | **5** | **6** |
| **Bootstrap** | 1 | 1 | 0,667 | 0,714 | 1 | 0,686 |
| **Tailwind** | 0,5 | 1 | 1 | 1 | 1 | 0 |
| **Foundation** | 1 | 0 | 0,667 | 0,571 | 0,8 | 0,748 |
| **Bulma** | 0 | 0,333 | 0,5 | 0,571 | 0,6 | 0,600 |
| **Skeleton** | 1 | 0,333 | 0 | 0 | 0 | 1 |

Pro výpočet celkového užitku konkrétní varianty je nutné sečíst hodnoty z odpovídajícího řádku matice, který reprezentuje danou variantu. Během tohoto sčítání musí být každá hodnota v řádku vynásobena normalizovanou vahou příslušného kritéria. Níže je uveden příklad, demonstrující postup výpočtu celkového užitku pro zvolenou variantu.

*Celkový užitek varianty 1:*

1 * 0,07 + 1 * 0,04 + 0,667 * 0,44 + 0,714 * 0,26 + 1 * 0,04 + 0,686 * 0,14 = **0,732**

*Tabulka 14 - Celkový užitek variant a jejich pořadí*

| | Celkový užitek variant | Pořadí |
|---|---|---|
| **Bootstrap** | 0,732 | 2. |
| **Tailwind** | 0,818 | 1. |
| **Foundation** | 0,660 | 3. |
| **Bulma** | 0,495 | 4. |
| **Skeleton** | 0,230 | 5. |

**Výsledné vyhodnocení**

Bylo provedeno porovnání různých UI frameworků z hlediska specificky stanovených kritérií. Celkový užitek pro každý framework byl vypočítán na základě standardizované kriteriální matice R, která byla konstruována s ohledem na ideální a bazální hodnoty pro jednotlivá kritéria.

Demonstrace této metody byla aplikována na pět různých UI frameworků – Bootstrap, Tailwind, Foundation, Bulma a Skeleton. Hodnoty ideální a bazální varianty byly vybrány na základě charakteristik každého kritéria, přičemž například pro kritérium velikosti frameworku (minimalizační kritérium) byla jako ideální hodnota zvolena nejmenší hodnota a jako bazální největší z hodnot získaných pro dané frameworky.

Na základě vypočítaných částečných užitků pro každé kritérium a framework byl určen celkový užitek každého z nich. Výsledky ukázaly, že Tailwind CSS dosáhl nejvyššího celkového užitku, což jej řadí na první místo mezi hodnocenými frameworky. Na druhém místě se umístil Bootstrap, následovaný Foundationem, Bulmou a na posledním místě Skeleton.

Tyto výsledky poskytují přehled o efektivitě a vhodnosti každého frameworku v kontextu stanovených kritérií a mohou sloužit jako důležitý nástroj pro vývojáře a designéry při výběru nejvhodnějšího UI frameworku pro jejich konkrétní projekt.

# 5 Výsledky a diskuse

## 5.1 Výsledky analýzy JavaScript frameworků

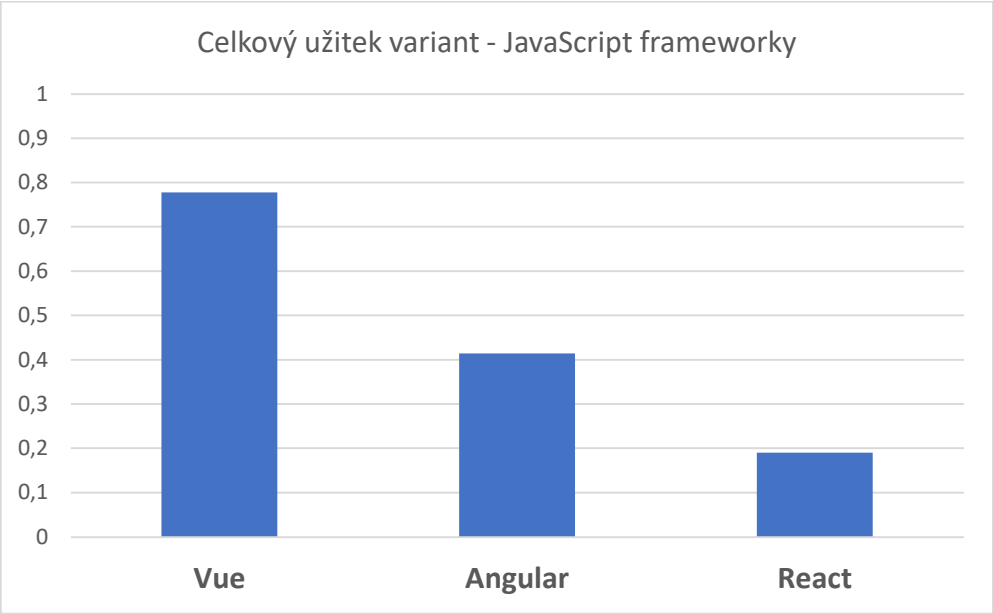
Prováděla se důkladná dvouetapová analýza JavaScriptových frameworků, v níž byly porovnány Angular, React, Vue, Svelte a Ember. Tyto JS frameworky byly vybrány na základě podobných vlastností a funkcí, kterými jednotlivé frameworky disponují. Za každým z těchto frameworků stojí specifická vývojářská komunita, která může být spravována velkou softwarovou společností (jako je Google nebo Facebook) nebo může být tvořena nezávislými vývojáři. V první etapě analýzy byly vyloučeny frameworky s nejmenším počtem aktivních členů na portále StackOverflow, nejmenším počtem vyhledávání prohlížečem Google Chrome a nejmenším počtem repozitářů založených na jednotlivých frameworcích. Kritéria byla také prověřena na českých pracovních portálech, jako jsou Jobs.cz a Freelance.cz, kde byla sledována poptávka po dovednostech spojených s těmito frameworky. Málo podporované frameworky mají totiž menší znalostní základnu mezi vývojáři a jsou využívány v méně projektech, což se odráží v počtu pracovních nabídek vyžadujících tyto technologie. V důsledku toho byly Ember a Svelte odstraněny z dalšího hodnocení. Pro Angular, React a Vue pak byla vytvořena ukázková webová aplikace pro předpověď počasí. Při vytváření této aplikace si mohl autor práce vyzkoušet tyto frameworky z hlediska vývoje. Dále byla provedena detailnější analýza s využitím vícekriteriálního přístupu a metody váženého součtu, kde byla stanovena a ohodnocena různá kritéria. Kromě kvantitativních kritérií zahrnovala analýza i kvalitativní kritéria jako dokumentaci nebo komponenty, která byla hodnocena s určitým stupněm subjektivity, zohledňujíc konkurenční povahu těchto frameworků.

Zajímavým aspektem této analýzy je, že výsledky hodnocení mohou být do značné míry ovlivněny povahou aplikace použité pro srovnání. V tomto případě byla použita relativně jednoduchá aplikace pro předpověď počasí, což mohlo přispět k tomu, že Vue vyšlo z analýzy jako framework s nejvyšším celkovým užitkem. Je důležité si uvědomit, že v případě složitější nebo rozsáhlejší webové aplikace by mohly být výsledky odlišné. Například, pokud by analýza zahrnovala vývoj robustnější aplikace, která by využívala komplexní architekturu a rozsáhlý ekosystém, je pravděpodobné, že by Angular měl větší užitek. Angular je známý svou pevnou strukturou a rozsáhlými funkcemi, které se mohou lépe hodit pro složité podnikové aplikace s vysokými nároky na udržitelnost a testovatelnost. Tento pohled naznačuje, že výběr nejvhodnějšího JavaScript frameworku může záviset na konkrétních

potřebách projektu a že výsledky takového srovnání mohou být různé v závislosti na typu a složitosti webové aplikace, která se vyvíjí.

Uvedený graf prezentuje celkový užitek pro každou variantu, který byl určen pomocí metody váženého součtu. Graf nejenže ukazuje specifické hodnoty užitku, ale také řadí jednotlivé frameworky podle výsledků získaných v rámci vícekriteriálního hodnocení.

*Tabulka 15 - Celkový užitek variant – JavaScript frameworky*




Na grafu je znázorněn celkový užitek pro Vue, Angular a React. Hodnoty celkového užitku jsou reprezentovány jako sloupce a jsou uspořádány svisle na osy y, která je rozdělena do intervalů po 0,1 až do maximální hodnoty 1. Vue má nejvyšší celkový užitek, který je 0,778. Angular má střední hodnotu celkového užitku, a to 0,414. React má nejnižší celkový užitek, který je 0,19.

## 5.2 Výsledky analýzy UI frameworků

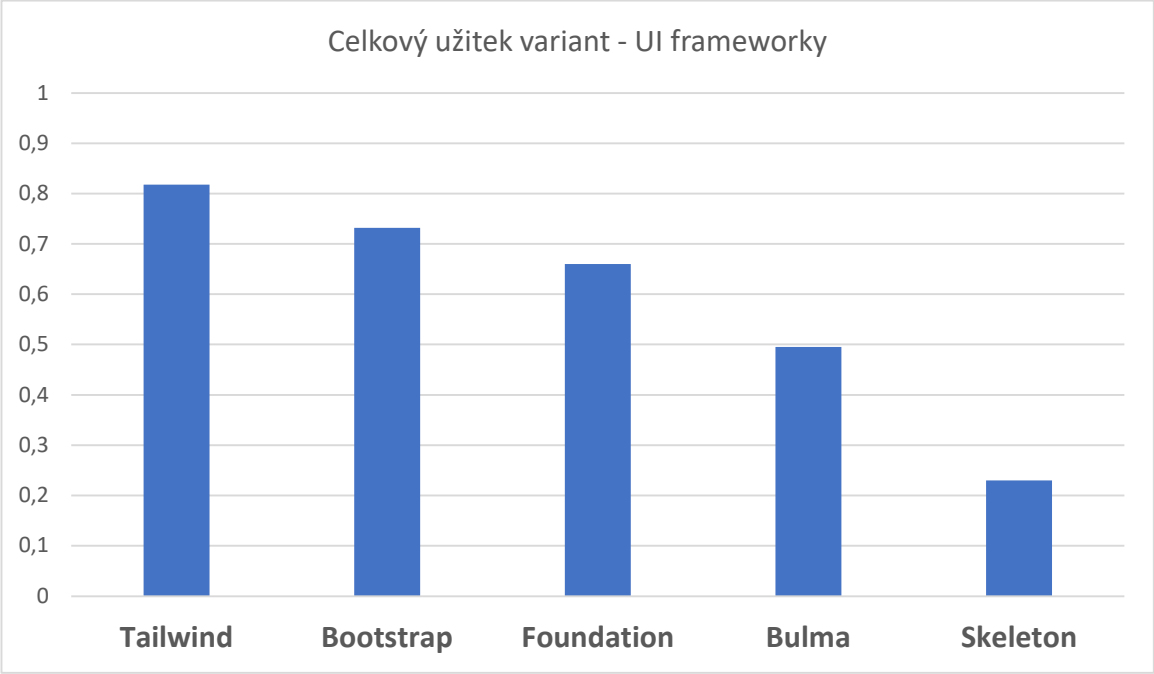
Výběr správného UI frameworku je zásadní pro vývoj responzivních webových aplikací na straně klienta. Pro tento účel byl sestaven přehled možností nabízených různými UI frameworky, následovaný prováděním hodnocení těchto frameworků pomocí metody váženého součtu. Analyzovány byly Bootstrap, Tailwind, Foundation, Bulma a Skeleton. V rámci hodnocení byla definována kritéria, kterým přiděleny body. Jediné kvantitativně měřitelné kritérium bylo velikost frameworku. Další kritéria, jako jsou mřížkové systémy, responzivní prvky, možnosti customizace a kompatibilita s prohlížeči, byla hodnocena spíše

subjektivně, založeno na analýze odborných pramenů od předních expertů na responzivní design.

Všechny zkoumané frameworky obsahují mřížkový systém a zlomové body, ale Bootstrap, Tailwind, Foundation a Bulma poskytují více funkcí než Skeleton, který je omezen pouze na základní mřížkový systém bez dodatečných responzivních nebo customizačních prvků. Výhodou Skeletonu je jeho malá velikost, pouhých 6 kB, což jej činí nejlehčím frameworkem mezi zkoumanými možnostmi.

Graf, který následuje, ilustruje výsledky vícekriteriální analýzy, ukazuje celkový užitek jednotlivých frameworků vypočítaný metodou váženého součtu a řadí je podle celkového skóre získaného v hodnocení.

*Tabulka 16 - Celkový užitek variant – UI frameworky*




V rámci vícekriteriální analýzy skončil Skeleton na posledním místě s celkovým užitkem pouhých 0,23. Bulma framework se umístil o něco lépe s hodnotou 0,495. Nejvyšší skóre a tím první místo získal Tailwind CSS s výrazně vyšším celkovým užitkem 0,818. Přestože Foundation a Bootstrap jsou obvykle považovány za konkurenty, bylo překvapivé vidět značný rozdíl ve skóre, přičemž Foundation skutečně předkládá mírně menší škálu funkcí oproti Bootstrapu, což se odrazilo ve sníženém bodovém ohodnocení v určitých kritériích.

# 6 Závěr

Cílem této diplomové práce bylo provést důkladné porovnání různých JavaScriptových a UI frameworků, které se využívají při vývoji webových stránek, s důrazem na front-end design a vývoj uživatelského rozhraní (UI). Práce systematicky zkoumala a porovnávala různé metodiky a postupy využívané v těchto frameworcích, poskytující tak ucelený přehled nástrojů a funkcí dostupných pro vývojáře.

V teoretické části byly představeny klíčové koncepty a postupy používané v CSS a JavaScript frameworcích. Byla provedena analýza, jak tyto technologie a metodiky mohou přispět k efektivnímu vývoji webových stránek, se zaměřením na aspekty, které ovlivňují design uživatelského rozhraní.

Praktická část práce se soustředila na aplikaci zjištěných teoretických poznatků prostřednictvím vývoje webových stránek s využitím vybraných frameworků. Tato část předvedla, jak teoretické principy mohou být uplatněny v praxi, a poskytla porovnání frameworků Angular, React a Vue na základě vyvinuté ukázkové aplikace. Porovnání bylo založeno na objektivně měřitelných kritériích a subjektivní hodnotící analýze, čímž byly získány ucelené informace o silných a slabých stránkách každého z nich.

Závěry této práce ukázaly, že každý framework nabízí jedinečné výhody a je vhodný pro různé typy projektů. Vue bylo hodnoceno jako nejvhodnější pro projekt použitý ve studii, nicméně bylo zdůrazněno, že volba frameworku by měla být vždy uzpůsobena specifickým požadavkům a cílům daného projektu.

Pro budoucí výzkum by bylo vhodné rozšířit analýzu na větší rozmanitost aplikací a zahrnout novější nebo méně známé frameworky, aby se poskytl ještě širší pohled na dostupné možnosti. Další výzkum by mohl také zahrnovat sledování vývoje a aktualizací jednotlivých frameworků, což by mohlo poskytnout hlubší porozumění jejich dlouhodobé udržitelnosti a adaptabilitě na měnící se trendy ve vývoji webových technologií.

V praktickém kontextu mohou výsledky této práce sloužit jako základ pro rozhodovací proces při výběru technologií pro nové projekty ve vývoji webových stránek a aplikací. S ohledem na rychlý vývoj v oblasti webdesignu a technologií je nezbytné, aby se vývojáři neustále vzdělávali a byli informováni o nejnovějších trendech a nejlepších postupech v oboru.

Tato práce tak poskytuje cenné poznatky, které mohou přispět k efektivnějšímu a informovanějšímu výběru technologií pro vývoj webových stránek, což je klíčové pro dosažení kvalitního a uživatelsky přívětivého výsledku.

# 7 Seznam použitých zdrojů


ALIF BUDIMAN, Muhammad. A Comprehensive Introduction to Tailwind CSS: Why Choose Tailwind CSS? In: *Medium* [online]. 2023 [cit. 2024-02-26]. Dostupné z: https://medium.com/@alifm2101/a-comprehensive-introduction-to-tailwind-css-36bc9cb81a1c.

An Introduction to SASS CSS: The CSS Pre-Processor: What Is SASS CSS?, 2009. Online. In: *Simplilearn.* Dostupné z: https://www.simplilearn.com/tutorials/css-tutorial/sass-css. [cit. 2024-02-27].

ARIS, B., 2023. *Hostinger. Hostinger Tutorials* [online]. [cit. 2024-02-26]. Dostupné z: https://www.hostinger.com/tutorials/dynamic-website.

BOOTSTRAP, 2024. *Browsers and devices.* Online. Dostupné z: https://getbootstrap.com/. [cit. 2024-03-06].

BOSE, Shreya. Top 5 CSS Frameworks for Developers and Designers: Foundation. In: *BrowserStack* [online]. 2011 [cit. 2024-02-26]. Dostupné z: https://www.browserstack.com/guide/top-css-frameworks

BULMA, 2024. *Documentation.* Online. Dostupné z: https://bulma.io/. [cit. 2024-03-06].

CANZIBA, Elvis, 2018. *Hands-On UX Design for Developers: Design, Prototype, and Implement Compelling User Experiences from Scratch: Design, Prototype, and Implement Compelling User Experiences from Scratch.* Birmingham, UNITED KINGDOM: Packt Publishing, Limited. ISBN 9781788624299

CARTER, Rebekah, 2022. A Brief History of Responsive Web Design. Online. In: *Web Designer Depot.* Dostupné z: https://www.webdesignerdepot.com/2022/06/a-brief-history-of-responsive-web-design/. [cit. 2024-02-28].

COREMANS, Chris, 2015. *HTML: A Beginner's Tutorial: A Beginner's Tutorial.* Vancouver, CANADA: Brainy Software. ISBN 9781771970198.

CSS Preprocessors Explained, 2020. Online. In: *FreeCodeCamp.* Dostupné z: https://www.freecodecamp.org/news/css-preprocessors/. [cit. 2024-02-27].

DESAI, Jemin, 2023. 5 Best Frontend Frameworks for Web Development in 2024: Key Considerations for Choosing the Top Front-End Frameworks 2024. Online. In: *POSITIWISE*. Dostupné z: https://positiwise.com/blog/best-front-end-frameworks#Key_Considerations_for_Choosing_the_Top_Front-End_Frameworks_2024. [cit. 2024-03-03].

DeviantArt, 2000. Online. In: *Web Design Museum.* Dostupné z: https://www.webdesignmuseum.org/web-design-history/deviantart-2000. [cit. 2024-02-28].

FEHERVARI, Zoltan, 2023. Most Popular Web Frameworks of the Year 2023: Top Insights: Ember JS. Online. In: *BLUEBIRD.* Dostupné z: https://bluebirdinternational.com/most-popular-web-frameworks/#:~:text=use%20Vue%20JS.-,Ember%20JS,-Ember%20JS%20is. [cit. 2024-02-28].

Freelance.cz, 2024. Online. Dostupné z: https://www.freelance.cz/. [cit. 2024-02-27].

FRIEDMAN, Vitaly, 2006. Responsive Web Design: What It Is And How To Use It: What Is Responsive Web Design? Online. In: *Smashing Magazine.* Dostupné z: https://www.smashingmagazine.com/2011/01/guidelines-for-responsive-web-design/. [cit. 2024-02-28].

FRISBIE, Matt, 2023. *Professional JavaScript for Web Developers.* Newark, UNITED STATES: John Wiley & Sons, Incorporated. ISBN 9781394193233. Dostupné také z: http://ebookcentral.proquest.com/lib/czup/detail.action?docID=30722280

GALANTE, Andrés, 2020. A Complete Guide to CSS Media Queries. Online. In: *Css-tricks.* Updated on Oct 19, 2022. Dostupné z: https://css-tricks.com/a-complete-guide-to-css-media-queries/. [cit. 2024-02-28].

Google Trends, 2024. Online. Dostupné z: https://trends.google.com/. [cit. 2024-02-27].

GOOGLE, 2024. *PageSpeed Insights*. Online. Dostupné z: https://pagespeed.web.dev/. [cit. 2024-03-03].

HARBER, Clive, 2023. *Vue. js for Jobseekers.* Los Angeles, INDIA: BPB Publications. ISBN 9789355518712. Dostupné také z: http://ebookcentral.proquest.com/lib/czup/detail.action?docID=30773847

HELD, Matthias, 2020. Co je síť pro doručování obsahu (CDN)? A je opravdu smysluplná?: Co je CDN? Online. In: *Raidboxes.* Aktualizováno dne 20.12.23. Dostupné z: https://raidboxes.io/cs/blog/hosting-performance/cdn-content-delivery-network/. [cit. 2024-03-06].

HOUŠKA, Milan, 2013. *Kurz: Ekonomicko matematické metody I - KS - ZS 13/14* [online]. [cit. 2024-02-27]. Dostupné z: https://moodle.czu.cz/course/view.php?id=805

How to Use Bootstrap Grid System in Your Website Design: Bootstrap Columns, 2017. Online. In: *BitDegree.* Updated Nov 14, 2019. Dostupné z: https://www.bitdegree.org/learn/bootstrap-grid-system. [cit. 2024-02-28].

HUMBLE, Charles, 2021. *All About Svelte, the Much-Loved, State-Driven Web Framework - The New Stack.* Online. TheNewStack. Dostupné z: https://thenewstack.io/all-about-svelte-the-much-loved-state-driven-web-framework/. [cit. 2024-02-27].

JAKOBUS, Benjamin, 2018. *Mastering Bootstrap 4: Master the Latest Version of Bootstrap 4 to Build Highly Customized Responsive Web Apps: Master the Latest*

*Version of Bootstrap 4 to Build Highly Customized Responsive Web Apps.* Birmingham, UNITED KINGDOM: Packt Publishing, Limited. ISBN 9781788838214.

Jobs.cz, 2024. Online. Dostupné z: https://www.jobs.cz/. [cit. 2024-02-27].

JULIVER, Jamie, 2006. What Is GitHub? (And What Is It Used For?): What is GitHub? Online. In: *HubSpot.* Dostupné z: https://blog.hubspot.com/website/what-is-github-used-for#what-github. [cit. 2024-02-28].

JUVILER, Jamie, 2024. What Is Fluid Design and How Is It Used on Websites?: Fluid Grids in Web Design. Online. In: *HubSpot.* Dostupné z: https://blog.hubspot.com/website/fluid-design. [cit. 2024-02-28].

KOISHIGAWA, Kris, 2021. A Brief History of Responsive Web Design. Online. In: *FreeCodeCamp.* Dostupné z: https://www.freecodecamp.org/news/a-brief-history-of-responsive-web-design/. [cit. 2024-02-28].

KROTOFF, Tanguys [@tkrotoff], 2024. Front-end frameworks popularity: GitHub repositories that depend on. Online. In: *GitHub Gist.* Dostupné z: https://gist.github.com/tkrotoff/b1caa4c3a185629299ec234d2314e190. [cit. 2024-02-28].

LARSEN, Rob, 2013. *Beginning HTML and CSS.* Newark, UNITED STATES: John Wiley & Sons, Incorporated. ISBN 9781118340288.

MARCOTTE, Ethan, 2010. Responsive Web Design. Online. In: *A List Apart.* Dostupné z: https://alistapart.com/article/responsive-web-design/. [cit. 2024-02-28].

MAURÉROVÁ, Veronika, 2023. Co je to responzivita webu a proč je důležitá. Online. In: *Unifer.* Dostupné z: https://unifer.cz/co-je-to-responzivita-webu-a-proc-je-dulezita/. [cit. 2024-02-29].

MICHÁLEK, Martin, 2013. CSS3 Media Queries. Online. In: *VZHŮRU DOLŮ.* Dostupné z: https://www.vzhurudolu.cz/prirucka/css3-media-queries. [cit. 2024-02-28].

MICHÁLEK, Martin, 2014. Co je to „Mobile First“? Ale doopravdy. Online. In: *VZHŮRU DOLŮ.* Dostupné z: https://www.vzhurudolu.cz/prirucka/mobile-first. [cit. 2024-02-29].

MONUS, Anna, 2023. *Popular CSS preprocessors with examples: Sass, Less, Stylus and more.* Online. RAYGUN. Dostupné z: https://raygun.com/blog/css-preprocessors-examples/#less. [cit. 2024-02-28].

PANCHAL, Jatin, 2021. Ember JS with its Pros and Cons: The Pros:. Online. In: *Rlogical.* Dostupné z: https://www.rlogical.com/blog/ember-js-with-its-pros-and-cons/. [cit. 2024-02-28].

RANJAN, Alok; SINHA, Abhilasha a BATTEWAD, Ranjit, 2020. *JavaScript for Modern Web Development.* Delhi, INDIA: BPB Publications. ISBN 9789389328738.

RIPPON, Carl, 2023. *Learn React with TypeScript: A Beginner's Guide to Reactive Web Development with React 18 and TypeScript.* Birmingham, UNITED KINGDOM: Packt Publishing, Limited. ISBN 9781804611050. Dostupné také z: http://ebookcentral.proquest.com/lib/czup/detail.action?docID=30391711

SHENOY, A, 2019. *Learning Bulma: Understand How to Develop Responsive, Mobile-first Websites Using This Impressive, Modern Framework: Understand How to Develop Responsive, Mobile-first Websites Using This Impressive, Modern Framework.* Apress. ISBN 9781484254820

SINGH, Daljit, 2023. Front-End Frameworks: A Comparison Guide: Compare popular front-end frameworks like React, Angular, and Vue.js to help developers make informed choice: Comparison. Online. In: *LinkedIn*. Dostupné z:

https://www.linkedin.com/pulse/front-end-frameworks-comparison-guide-compare-popular-daljit-singh/. [cit. 2024-03-03].

SKELETON, 2024. *A dead simple, responsive boilerplate.* Online. Dostupné z: http://getskeleton.com/. [cit. 2024-03-06].

Stack Overflow Trends, 2024. Online. In: *Stackoverflow.* Dostupné z: https://insights.stackoverflow.com/trends?tags=svelte%2Creactjs%2Cvue.js%2Cember.js%2Cangular. [cit. 2024-02-27].

ŠUBRT, Tomáš, 2011. *Ekonomicko-matematické metody.* Plzeň: Vydavatelství a nakladatelství Aleš Čeněk. ISBN ISBN978-80-7380-345-2.

TAILWINDCSS, 2024. *Browser Support.* Online. Dostupné z: https://tailwindcss.com/. [cit. 2024-03-06].

TERRA, John, 2023. Breaking Down CSS and CSS3 Differences: What is CSS? Online. In: *Simplilearn.* Last updated on Feb 23, 2023. Dostupné z: https://www.simplilearn.com/difference-between-css-and-css3-article. [cit. 2024-02-28].

The 2022 State of JS: About, 2022. Online. In: *State of JavaScript 2022.* Dostupné z: https://2022.stateofjs.com/en-US/about/. [cit. 2024-02-27].

TRIPATHI, Satyam, 2024. Bulma CSS Framework: Getting Started Guide: What is Bulma CSS? Online. In: *LAMBDATEST*. Dostupné z: https://www.lambdatest.com/blog/bulma-css-framework/. [cit. 2024-03-06].

UNADKAT, Jash, 2011. Mobile First Design: What it is and How to Implement it. Online. In: *BrowserStack.* Dostupné z: https://www.browserstack.com/guide/how-to-implement-mobile-first-design. [cit. 2024-02-29].

What is Angular?, 2023. *Angular* [online]. Last reviewed on Tue Aug 15 2023 [cit. 2024-02-26]. Dostupné z: https://angular.io/guide/what-is-angular

WOLF, Jürgen, 2023. *HTML and CSS: The Comprehensive Guide: The Comprehensive Guide.* Boston, UNITED STATES: Rheinwerk Publishing. ISBN 9781493224234.

Yahoo!, 1994. Online. In: *Web Design Museum.* Dostupné z: https://www.webdesignmuseum.org/web-design-history/yahoo-1994. [cit. 2024-02-28].

# 8 Seznam obrázků, tabulek, grafů a zkratek

## 8.1 Seznam obrázků

## 8.2 Seznam tabulek



## 8.3 Seznam grafů



## 8.4 Seznam použitých zkratek

JS – JavaScript
UI – User Interface
HTML – Hypertext Markup Language
CSS – Kaskádové styly

# Přílohy

Odkazovaný seznam příloh