**Univerzita Karlova**

**Pedagogická fakulta**

Katedra informačních technologií a technické výchovy

# Bakalářská práce

Aleksandar Golotvinov

## Analýza, srovnání a aplikace současných webových frontendových knihoven

Analysis, comparison and application of the current web frontend frameworks

Praha 2021

Vedoucí práce: PhDr. Josef Procházka, Ph.D.

**Prohlášení:**

Prohlašuji, že jsem bakalářskou práci vypracoval samostatně, že jsem řádně citoval všechny použité prameny a literaturu a že práce nebyla využita v rámci jiného vysokoškolského studia či k získání jiného nebo stejného titulu.

V Praze, dne 12. července 2021

Aleksandar Golotvinov

**Poděkování:**

Rád bych touto cestou vyjádřil poděkování panu PhDr. Josefu Procházkovi, Ph.D. za jeho cenné rady a trpělivost při vedení mé bakalářské práce.

**Abstrakt**

První část této bakalářské práce se věnuje tvorbě, tedy vývoje a grafickému návrhu, webových stránek a aplikací z teoretického hlediska. Problematiku webdesignu zkoumá spíše z pohledu současnosti, avšak historický vývoj je zde také zahrnut. Praktickou část práce představuje tvorba celkem čtyř funkčních prototypů webových stránek na základě kterých je provedeno detailnější srovnání jednotlivých knihoven. Jeden z prototypů je tvořen bez užití jakýchkoliv knihoven a ostatní tři jsou realizovány pomocí vybraných CSS frameworků.

**Klíčová slova**

web, grafika, rozložení, vývoj, návrh, framework, knihovna, HTML, CSS

**Abstract**

The first part of this bachelor thesis is devoted to the creation, i.e. development and graphic design, of websites and applications from a theoretical point of view. It explores the issue of web design from a more contemporary perspective, but historical development is also included. The practical part of the thesis is the creation of a total of four functional prototypes of websites on the basis of which a detailed comparison of the different libraries is made. One of the prototypes is created without using any libraries and the other three are made using the selected CSS frameworks.

**Keywords**

web, graphics, layout, development, design, framework, library, HTML, CSS

# Obsah

# Úvod

Frontendové knihovny představují jakýsi průnik mezi návrhem a vývojem, jsou v nich obsaženy oba tyto okruhy. Cílem této práce je především srovnat populární webové frontendové knihovny, poukázat na jejich klady a zápory, porovnat je mezi sebou a porovnat je i s přístupem bez použití obdobných pomocných nástrojů, tedy tvorba webu bez použití jakýchkoliv CSS knihoven.

Dalším dílčím cílem je popsat požadavky, pravidla a základní koncepty při tvorbě webových stránek a aplikací. Ať už se jedná o principy při tvorbě designu či při psaní kódu a tvorbě jednotlivých elementů na stránce.

Praktická část práce je řešena formou tvorby vzorového web designu a čtyř funkčních prototypů webů. Tři z nich jsou vyrobeny pomocí vybraných frontendových knihoven, a jeden je vyroben bez použití knihovny. Následně jsou webové stránky srovnány z hlediska rychlosti načítání, objemu kódu a celkové kvality výsledného projektu.

Tato práce nebude zaměřena na komplexní frontendové knihovny jako jsou např. React, Angular, Vue, Nette, Laravel, Django a další, jejichž účel je tvorba celých webových aplikací. Hlavní oblast zaměření budou tvořit knihovny, které jsou často označovány jako CSS knihovny, sloužící primárně k tvorbě vizuální podoby webových stránek, k nakódování grafického návrhu a jeho přeměně z „prostého obrázku“ na funkční prototyp. Většina těchto frontend frameworků neobsahuje žádný JavaScript (zde jsou výjimky např. Bootstrap, Foundation a další) či PHP nebo jiné skriptovací jazyky.[1]

[1] CSS frameworky typu Boostrap nebo Tailwind však mohou být snadno implementovány např. do Reactu či Angularu jako doplňky nebo pomocné pluginy. Příkladem konkrétního řešení je pak ng-bootstrap (https://ng-bootstrap.github.io/) nebo React-Bootstrap (https://react-bootstrap.github.io/). Instalace probíhá nejčastěji pomocí nějakého správce balíčků, jeden z nejpoužívanějších je npm (https://www.npmjs.com/).

# 1 Historický vývoj tvorby webdesignu

Pro lepší pochopení problematiky současných webových frameworků je nutné se podívat na historický kontext, ze kterého je možno vidět vývoj webdesignu a získat řádnější porozumění pro důvody, kvůli kterým vznikly CSS knihovny.

## 1.1 Web 1.0

Termín Web 1.0 vznikl až poté co se v roce 2004 konala první konference Web 2.0 Summit. S názvem přišel Dale Dougherty, zaměstnanec firmy O'Reilly Media, která akci organizovala. Jeho cílem bylo vymyslet chytlavé jméno, ale žádný další význam tomuto pojmenování nedal. Definice termínů Web 1.0 a Web 2.0 přišly později, avšak dodnes jsou nepřesné.

Termín Web 1.0 se neváže na časové období, nebo na použité technologie (a totéž platí i o Webu 2.0). Obecně řečeno se jedná o počátek WWW, kdy webové stránky byly statické, neinteraktivní a často postavené pomocí proprietárních technologií. Je to označení pro první generaci webů. (Strickland, 2021)

### 1.1.1 Počátek WWW, první webová stránka

Vůbec první webovou stránku vytvořil v roce 1991 Tim Berners-Lee, když pracoval pro CERN. Potřeboval vymyslet jednoduchý a rychlý systém pro sdílení souborů mezi jednotlivými zaměstnanci. A tím byl právě projekt World Wide Web (WWW), v rámci kterého Berners-Lee vytvořil (byl hlavním autorem) i značkovací jazyk HTML, přenosový protokol HTTP, který slouží pro komunikaci s webovými servery a specifikace URL. Byly položeny základy webového světa, avšak v tu dobu ještě nikdo netušil, že se rozroste do rozměrů, kterých nabývá v dnešní době.

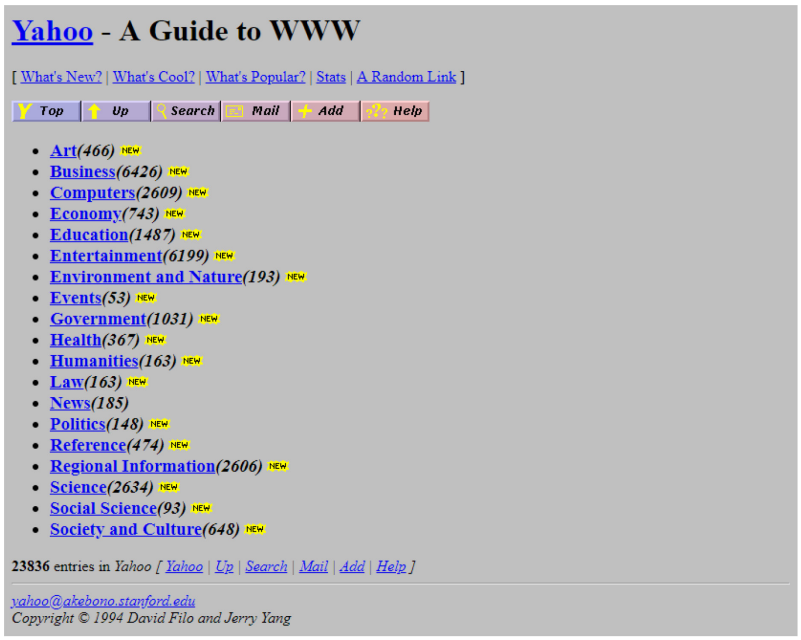
Samotná stránka je z pohledu současných webů nesmírně jednoduchá, dokonce nevyhovující. Obsahuje však jeden z nejzákladnějších prvků webu – hypertextový odkaz. V dnešní době může jako hypertextový odkaz fungovat např. i obrázek, tlačítko, multimediální prvek a další. V roce 1991 ale tuto vlastnost mohl mít pouze text. Struktura webu byla primitivní: Tvořil ji jediný HTML

soubor, který odkazoval na několik dalších, které vypadaly prakticky stejně. Měnil se tedy pouze obsah, struktura zůstala stejná.

Po grafické stránce zde nenajdeme nic zajímavého. Pozadí je tvořeno výchozí barvou webového prohlížeče, text taktéž přebírá výchozí hodnoty browseru. Nic z toho však není překvapující, bereme-li v potaz účel projektu, zařízení, pro který byl web určen a dostupné technologie. Monitory měly většinou nízké rozlišení (VGA, 640x480) a byly monochromatické (barevné obrazovky sice existovaly, ale stále nebyly rozšířené). Výpočetní výkon byl nesrovnatelně nižší, cílová skupina se skládala pouze z několika desítek lidí, kolegů T. B. Leeho. Neexistovaly jazyky jako PHP nebo Javascript, které by umožňovaly dodat webům dynamičnost, proto se v době vzniku prvního webu můžeme bavit pouze o tzv. statických webových stránkách, tedy o takových stránkách, jejichž obsah se nemění (ani na základě uživatelského požadavku, ani na příkaz ze strany serveru). Ty přichází o pár let později.

### 1.1.2 První verze webů

Velká část internetových technologií, které používáme dodnes, vznikla několik málo let poté, co byla spuštěna první webové stránka. Ačkoliv web zpočátku existoval skoro exkluzivně v akademickém prostředí, již necelé dva roky po jeho zrodu se začaly objevovat stránky dostupné široké veřejnosti (což byl ale stále velmi úzký okruh lidí, který měl přístup k internetu a potřebné znalosti na to s počítačem pracovat). Typickým příkladem je Yahoo!, který vytvořili Jerry Yang a David Filo, studenti Stanfordovi univerzity. Sloužil jako jakýsi rejstřík či katalog WWW a obsahoval odkazy na jiné stránky z různých oblastí zájmů. Později k němu přidali i funkci hledání (jeden z prvních webových vyhledávačů na světě).




*Obrázek 1 Stránka Yahoo! v roce 1994 (Zdroj: Web Design Museum)*

Již na první pohled je jasné, že se jedná o jednoduchý HTML dokument, který obsahuje hlavně hypertextové odkazy a prostý text formou paragrafů a nadpisů. Objevuje se tam i pár GIF obrázků (horní menu a štítek „NEW“). Jak již bylo zmíněno, jedná se o typický příklad z tohoto období. Naprostá většina webových projektů z této doby (rok 1992–1995) vypadala zhruba takto a sdílela podobnou strukturu a podobu. Jednoduché, většinou jednobarevné pozadí, nadpisy a paragrafy černé barvy, hypertextové odkazy modré a podtržené, celkový layout uzpůsoben pro monitory s poměrem stran 4:3.

Je nesmírně podstatné vždy brát v potaz, že bylo toto období (90. léta 20. stol.) jakýsi „divoký západ“ pro web. Internetový svět byl ještě ve svých počátcích, technologie byly relativně omezující (výpočetní výkon, zobrazovací prostředky, dostupný obsah, rozsah, vzdělanost uživatelů v problematice atd.). Weboví vývojáři a designeři (tehdy to byli vlastně ti samí lidé, oddělení složek grafiky a vývoje přišla později) byli inovátoři a experimentovali. Neexistovalo mnoho zdrojů literatury na téma „jak by měl dobrý web vypadat“. Neexistoval koherentní designový jazyk, který by určoval jakým směrem by se měly weby vydávat. To mělo za důsledek, že byly webové stránky velice rozmanité po grafické stránce, avšak po stránce kvalitativní by v dnešní době vůbec neobstály. Dnešní uživatel již má nějakou základní představu kde by se měla nacházet hlavní navigace webu, logo, hlavní obsah anebo patička s doplňujícími informacemi.

Očekává se i to, že bude web splňovat jisté standardy přístupnosti (accessibility) – o tom bude detailnější popis v kapitole o tvorbě webů dnes.

Samotný kód byl pochopitelně také velice jednoduchý, tedy alespoň do doby, kdy v roce 1995 vznikly skriptovací jazyky PHP a JavaScript a v roce 1996 vyšla první verze kaskádových stylů (CSS). Samotnou implementaci těchto jazyků do konkrétních projektů pak začínáme vnímat okolo roku 1997, kdy již začala postupně vznikat vývojářská komunita, která tyto technologie aktivně využívala.

### 1.1.3 Zrod webových technologií

Během 90. let postupně vznikaly nové technologie, nástroje a možnosti, jak obohatit tuto ještě relativně novou disciplínu v informatice. V následujících podkapitolách jsou stručně popsány ty nejdůležitější z nich.[2]

#### HTML

Hypertext Markup Language, neboli HTML, je základním stavebním kámenem všech webových stránek. Jak již bylo dřív v práci zmíněno, jeho tvůrcem je Tim Berners-Lee. Jedná se o značkovací jazyk (nikoliv programovací), který tvoří základní strukturu každé webové stránky. Pomocí HTML určujeme jaké elementy se na stránce nachází, jakého jsou typu a kde se vyskytuje.[3] Často se o něm říká, že je to kostra webu.

První verze, která vyšla v roce 1990, byla velmi omezená. Šlo využít pouze malý počet základních tagů (značek) jako např. h1 (odstavec nejvyššího řádu), p (paragraf, tok textu), ul (unordered list = neuspořádaný seznam), a (anchor, hypertextový odkaz) a pár dalších.

Verze 2.0 byla vydána v listopadu 1995 a obsahovala spoustu novinek. Mimo jiné byly přidány i tagy pro určení typu dokumentu, pro hlavičku, tělo, formuláře, obrázky a další.

Standard HTML 3.2 vyšel v roce 1997 (verze 3.0 se nikdy standardem nestala, protože byla moc komplikovaná). Stejně jako svůj předchůdce přinesl

[2] O PHP nebude řeč, ačkoliv se jedná o podstatnou složku z hlediska webových technologií, a to z toho důvodu, že nesouvisí s frontendem či s designem / grafikou.
[3] Pořadí prvků v kódu však nemusí nutně odpovídat jejich vizuální hierarchii. O tom se pak stará CSS např. pomocí Flexboxu, Grid systému atd.

řadu novinek a vylepšení z nichž ty nejpodstatnější jsou tagy na centrování obsahu, obecný tag div a značky na vložení stylů a skriptů.

Poslední verze jazyka, která vyšla v 90. letech (konkrétně v roce 1997) je HTML 4.0, která jako první klade důraz na sémantiku a na rozdělení sémantiky a struktury, což ve výsledku znamená, že je dobré oddělit obsahovou HTML část a styly (propojit s externím souborem) a používat značky, které jsou určené pro daný typ obsahu.

### JavaScript

V prosinci roku 1995 byla vydána první verze jazyka JavaScript, jehož tvůrcem je americký programátor Brendan Eich. Jde o objektově orientovaný skriptovací jazyk, který pracuje na straně klienta (jsou však řešení i na straně serveru jako např. NodeJS). Je využíván nejčastěji k tvorbě a ovládání uživatelských rozhraní a přeměně obsahu webových stránek a také slouží k tvorbě dynamických částí webu. V dnešní době je hojně používán v rámci různých frameworků jako jsou např. React, Angular, Vue nebo Express či knihoven jako je jQuery. V roce 2020 byl celkově nejvíce používaným programovacím jazykem na světě. (JetBrains, 2020)

### CSS

Kaskádové styly (CSS) jsou jazyk, který určuje pozici, vzhled, rozložení, velikost a další vizuální vlastnosti elementů v dokumentech psaných v nějakém značkovacím jazyce (HTML, XHTML, XML a další). První verze CSS vyšla na konci roku 1996. S příchodem této technologie již mohly mít webové stránky podstatně zajímavější, pestřejší ale i složitější design. Kaskádové styly umožňují jednoduché formátování textu, obrázků a multimediálních prvků na stránce a ve verzi CSS3 i tvorbu animací.

Kaskádové styly jsou tím nejdůležitějším prvkem při tvorbě vizuální části webu a v době svého vzniku byly pomyslným prvním krokem při separaci obsahu a stylů.

## 1.2 Web 2.0

Jak již bylo zmíněno, termín Web 2.0 vznikl dřív než Web 1.0. Jedná se neoficiálně o druhou generaci webových stránek, které, oproti té první, začaly

nabízet interaktivitu a dynamičnost a byly alespoň částečně postavené pomocí open source technologií. Uživatel již mohl nějakým způsobem ovlivňovat co se na webu děje. Obsah už není nutně pouze pro čtení.

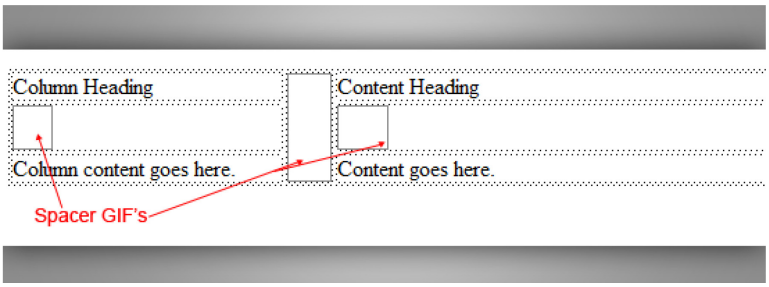
Všechny již zmíněné nástroje a technologie, které byly již k dispozici, umožnili to, aby vznikaly webové stránky stále rozmanitější a designově zajímavější. Často se vyskytují křiklavé barvy, animované GIFy, velké množství fotografií, obrázků a elementy, které jsou pouze ornamentní, neslouží žádným praktickým účelům. Pozadí bývají jak jednoduchá, tak i rušná a často také animovaná. Na některých webech se už začíná rýsovat dnes již známá základní struktura: nahoře hlavička, která obsahuje logo a hlavní navigaci, následuje hlavní obsah a případné vedlejší navigační prvky, a dole patička s doplňujícími informacemi. Jednotný, koherentní designový jazyk nicméně stále neexistoval. Sémantice a přístupnosti stále není věnována pozornost, v popředí je estetika a snaha zaujmout a předvést něco nového, něco originálního.


*Obrázek Oficiální webové stránky rappera Eminema v roce 1999 (Zdroj: Web Design Museum)*

Složitější design samozřejmě představoval větší výzvu pro kodéry. Zatím stále neexistovaly nástroje pro přesné pozicování, white space[4] nebo

[4] White space (také whitespace, negative space) je neobsazený, volný prostor mezi jednotlivými elementy. U webdesignu to jsou např. margin, padding, gutter a další.

vícesloupcové či jakkoliv komplexnější rozložení. Proto se používaly funkce, které tomu nebyly určeny. Pro tvorbu layoutu webu se využívaly tabulky (tzv. tabulkový design). Jednotlivé elementy se pak vkládaly do sloupců, řádků a buněk tabulky. Tabulky se používaly se i pro rozdělení obrázků na více částí což umožnilo na stránku de facto vložit celý obrázek a následně určit které části budou fungovat jako odkaz. Další „hack“ při převodu designu na kód byly tzv. spacer GIF-y. Jsou to malé, průhledné obrázky, sloužící k tvorbě white space, tedy k přidání „volného prostoru“. V dnešní době se toto řeší pomocí paddingu, či marginu.




*Obrázek 2 Spacer GIF-y v tabulkovém layoutu (Zdroj: WebFX)*

### 1.2.1 W3C

S rostoucím zájmem i množstvím webů, které vznikalo bylo potřeba celý proces nějakým způsobem regulovat, standardizovat a posilovat. Proto v roce 1994 založil nám již známý Tim Berners-Lee World Wide Web Consortium (W3C). Mezinárodní konsorcium, které vyvíjí a udržuje webové technologie, poskytuje osvětu a vzdělání a nabízí i validátor kódu, který je pro účely kvalitního webového vývoje nesmírně podstatný. Validátor kódu je nástroj, který kontroluje kód a ukazuje na chyby. Za chybné části pak považuje ty části kódu, které jsou v rozporu se specifikací. „Upozorní vás na nesprávné použití prvků, když prvky zařadíte tam, kde nemají být, když vynecháte povinné atributy, uvedete nesprávné hodnoty atributů a podobné věci.“ (Goldstein, 2011) Takový nástroj bývá také často součástí IDE a nemusí být jenom určen pro HTML a CSS nýbrž i pro jiné značkovací a programovací jazyky.

### 1.2.2 Stránky postavené na technologii Flash

Na přelomu tisíciletí se začaly objevovat první webové stránky postavené na technologii Flash a jejich počet velmi rychle rostl. Adobe Flash (do roku 2005 Macromedia Flash) je multimediální vývojářská platforma pomocí které lze vytvářet animace, animované a interaktivní webové aplikace, hry, mobilní aplikace, webové přehrávače a další. Umožňuje práci s textem, vektorovou grafikou, rastrovou grafikou, videem, zvukem, pohybem myší, inputem z klávesnice, mikrofonu a webkamery. Oproti tradičním možnostem webového vývoje v této době nabízel Flash mnohem více funkcí a způsobů jak naprogramovat poutavou a interaktivní webovou prezentaci. Vznikaly stránky nabízející uživatelům vysoce stylizovaná prostředí, která obsahovala mix multimediálních prvků. Animace a zvuky po najetím myší na prvku (hover efekty), drag and drop sekce, hry přímo vložené do webu, animované galerie, personalizované audio a video přehrávače. Toto je pouze stručný výčet komponent, které se díky Flashi začaly objevovat. V souvislosti s tím se objevil termín rich web application (někdy také rich internet application), který popisuje webovou aplikaci, která sdílí některé charakteristiky desktopových aplikací jako např. strukturu uživatelského rozhraní, drag and drop, WYSIWYG editace a další.

I v tomto období přetrvává trend experimentování a upřednostňování formy nad funkcí. Separace obsahové, stylové a programové části možná není z důvodu architektury flashových aplikací.


*Obrázek 3 Flash web Conspiracy Games z roku 2002 (Zdroj: Web Design Museum)*

Byla to skutečně revoluční technologie, která byla rychle adoptována webovými vývojáři a byla podporována všemi velkými internetovými prohlížeči. Přinesla mnoho novinek a kompletně změnila způsob tvorby webových stránek. V dnešní době je Flash (SWF formát) pro webový vývoj již zastaralý a nepoužívaný. Internetové prohlížeče, postupně odstraňovaly podporu a 31. prosince 2020 ukončilo Adobe podporu pro Flash player. Flash aplikacím je vyčítáno mimo jiné jejich bezpečnost, resp. jejich díry v zabezpečení, rozbíjení funkce tlačítka „Zpět“ v prohlížeči a proprietární licencování. K úbytku na popularitě se také podílely HTML5, CSS3 a JavaScript, pomocí kterých již šlo vytvořit podobné aplikace efektivněji.

## 1.3 Nástup současných trendů

V roce 2007 byl představen první iPhone. Tímto vznikl jakýsi pomyslný milník kdy mobilní zařízení začala hrát ve světě webů významnou roli. Prohlížení webových stránek na telefonech bylo možné dávno předtím, ale s nástupem smartphonů a rozšířením technologií jako 3G a WiFi se tato činnost stala běžnou záležitostí a počet uživatelů, kteří si prohlíželi stránky z mobilních zařízení, od té doby stále stoupal.

Diverzifikace rozlišení a poměrů stran u monitorů měla podobný dopad na web jako smartphony: Bylo potřeba vytvářet takové webové stránky, které by vypadaly dobře na více různorodých zařízení. Mezi požadavky klientů na vývojáře přibyla další funkcionalita. Tou byla právě responzivita, tedy schopnost webu se „transformovat“ a vypadat a fungovat dobře na všech dostupných a používaných zařízeních za cílem oslovit co největší počet uživatelů (a z hlediska byznysu potenciálních klientů). V roce 2012 vydalo W3C první specifikaci pro Media Queries, CSS modul, který umožňuje adaptovat web pro různá zařízení a rozlišení.

S odstupem Flashe se preferovanými technologiemi opět staly HTML (od roku 2014 ve verzi 5) a CSS3 spolu s JavaScriptem pro front end vývoj.[5]

Webové projekty rostly, byly čím dál tím rozsáhlejší a složitější a kód byl stále delší. Proto se z důvodu přehlednosti a dlouhodobé udržitelnosti začaly rozdělovat jednotlivé složky. Obsah patří do HTML, stylování skoro výhradně pomocí CSS, které je ve zvláštním souboru. Skripty a další moduly a doplňky taktéž – do zvláštních souboru a složek.

Začala se řešit i otázka přístupnosti, tedy přizpůsobení obsahu a vizuální části lidem se zrakovými, sluchovými či motorickými potížemi. Rozrůstala se myšlenka zpřístupnit web co nejvíce lidem.

Některé z velkých společností usilovaly o zavedení jednotných designových trendů, které by sjednotily strukturu a částečně i vzhled webových stránek tak, aby se návštěvník mohl snadno a rychle orientovat na různých webech. Příkladem je Microsoft a jeho Flat design, který představil světu v roce 2009 na stránkách Zune. O pár let později, v roce 2014, pak představil Google svůj Material design. Odlišností je celá řada, ale podstatné je, že oba tyto směry spojuje klíčový faktor: jednoduchost a čisté, odlehčené prostředí, ve kterém vynikne obsah.

### 1.3.1 První (moderní) CSS knihovny

S narůstající komplexitou a stále vyššími nároky na kvalitní webovou stránku, vznikla potřeba vytvořit nástroje, které by tvorbu frontendu po grafické stránce vývojářům usnadnili. Tím byly právě CSS frameworky, mezi jejichž první

[5] Ačkoliv v období 2000–2010 byla velká část webů postavená na Flash technologii, neznamená to, že „tradičně“ vytvořené stránky pomocí HTML, CSS, PHP a JavaScriptu neexistovaly.

úspěšné a hojně rozšířené zástupce můžeme zařadit Blueprint CSS a 960 Grid System. Oba projekty vznikly okolo roku 2010 a měly za cíl poskytnout webdesignerům pomůcky, kterými by urychlily a usnadnily proces tvorby prototypů i finálních produkčních verzí webových projektů. Ačkoliv předtím podobné prostředky existovaly, tyto se svojí myšlenkou, strukturou a provedením velice podobají dnešním řešením. Jsou to knihovny, které kladou velký důraz na mřížkové rozložení, konzistencí mezi prohlížeči, modulárním přístupu a poskytnutí předpřipravených komponent pro koherentní design.

**Blueprint CSS**

Blueprint CSS je CSS knihovna, obsahující snadno upravitelný grid systém, základní typografické úpravy pro dobře čitelné písmo, CSS reset[6], styly pro tisk, skript pro minimalizaci a komprimaci výsledného CSS, skript na W3C validaci kódu a „hacky“ pro Internet Explorer. Její autorem je Josh Clayton, skripty jsou psané v Ruby, a kód je stále k dispozici na GitHubu, ačkoliv byl vývoj knihovny ukončen před více než deseti lety. Blueprint je modulární, což znamená, že jednotlivé komponenty jsou rozděleny do souborů (grid.css, forms.css, print.css, typography.css, …) a je tudíž možné do projektu zakomponovat pouze ty části, které jsou potřeba.

Jednoznačně nejužitečnější část knihovny je flexibilní mřížkový layout, který umožňuje rozprostřít jednotlivé elementy na webu do sudého počtu sloupců (nejčastěji 18 či 24). Maximální šířka kontejneru je 950px, je vycentrovaný, a je možné do něho dát na jeden řádek maximálně 24 sloupců. Vše je řešeno absolutními hodnotami, od šířky sloupce, přes mezery mezi nimi až po nastavení velikosti jednotlivých objektů v gridu, které přebírají šířku sloupce či sloupců jim náležící.

```
/* Úprava šířky jednotlivých sloupců */
.span-1 {width: 30px;}
.span-2 {width: 70px;}
.span-3 {width: 110px;
...

/* Klasy na posunutí objektů do předchozího a do následujícího sloupce */
...
.pull-6 { margin-left: -240px; }
.pull-7 { margin-left: -280px; }
```

[6] Viz podkapitola 2.7 Cross-browser compatibility

```
.pull-8 { margin-left: -320px; }
...

...
.push-22 { margin: 0 -880px 1.5em 880px; }
.push-23 { margin: 0 -920px 1.5em 920px; }
.push-24 { margin: 0 -960px 1.5em 960px; }
```

### 960 Grid System

960 Grid Systém se liší nejen od Blueprint CSS ale i od všech frameworků, které jsou k dispozici dnes. Název samotný poukazuje na 3 základní koncepty tohoto projektu. Je to knihovna, která využívá grid systém, používá pro tu dobu standardní šířku 960px a systém implikuje, že se nejedná pouze o několik pomocných CSS souborů. Tento framework totiž obsahuje také šablony pro Sketch, Photoshop, Fireworks, Illustrator, Inkscape, Corel Draw, GIMP a mnohé další, ve kterých je připraven pracovní dokument s 12sloupcovým gridem, kde je možné si design přesně navrhnout. Při stažení, a v oficiálním repozitáři, jsou k dispozici i soubory pro tisk, šablony a dokumentace. Krom standardních 12 sloupců nabízí 960 Grid Systém i verze 16sloupcové a 18sloupcové.

Posouvání a pozicování prvků je řešeno stejně jako u Blueprint CSS, taktéž obsahuje reset defaultních stylů prohlížeče a dokonce disponuje styly pro RTL[7] layouty. Jedná se tedy o komplexní designový systém pro tvorbu grafické podoby webů.

[7] RTL je zkratka pro right-to-left a označuje jazyky ve kterých se píše zprava doleva jako např. arabština a hebrejština.

# 2 Současné požadavky na webovou stránku z hlediska designu

Vzhled webu musí být komplementární obsahu, měl by návštěvníkovi sdělit jasně a zřetelně, jaké je zaměření stránky, aniž by odváděl jeho pozornost nebo ho zbytečně rušil. Toho lze docílit různými technikami, nejefektivnější z nichž je volba správné barevné palety.

Současné požadavky na kvalitní vizuální styl pochází z principu UCD, User-Centered Design. Ten říká, že středem vývoje by měl být samotný uživatel. Design by měl být tedy vyhovující jeho očekáváním, požadavkům a schopnostem mu porozumět. Je to filozofie, která říká vývojářům a designerům, že je produkt určen uživatelům a je podstatné ho vytvořit tak, aby cílové skupině sloužil co nejlépe. Vývojáři a grafici vnímají projekty na kterých pracují odlišně od svých klientů či zadavatelů, což je zcela normální. Proto se obecně doporučuje dělat user testy ve více fázích tvorby webu a sledovat zpětnou vazbu a jak aplikaci vnímají ti, kteří se na její tvorbě přímo nepodílí.

Praktickou aplikaci UCD principu při tvorbě webdesignu můžeme implementovat pomocí několika konceptů a myšlenek. Mezi ně patří např.:

- Vytvořit plně responzivní design, který bude plnohodnotně uplatněn na všech cílových zařízeních
- Vyřešit základní problémy přístupnosti – kontrast, velikost a čitelnost písma, optimalizace pro pomocná zařízení a další
- Vytvořit smysluplný layout s přehlednou strukturou a jasnými navigačními prvky
- Odebrat veškeré zbytečné a rušivé elementy v aplikaci

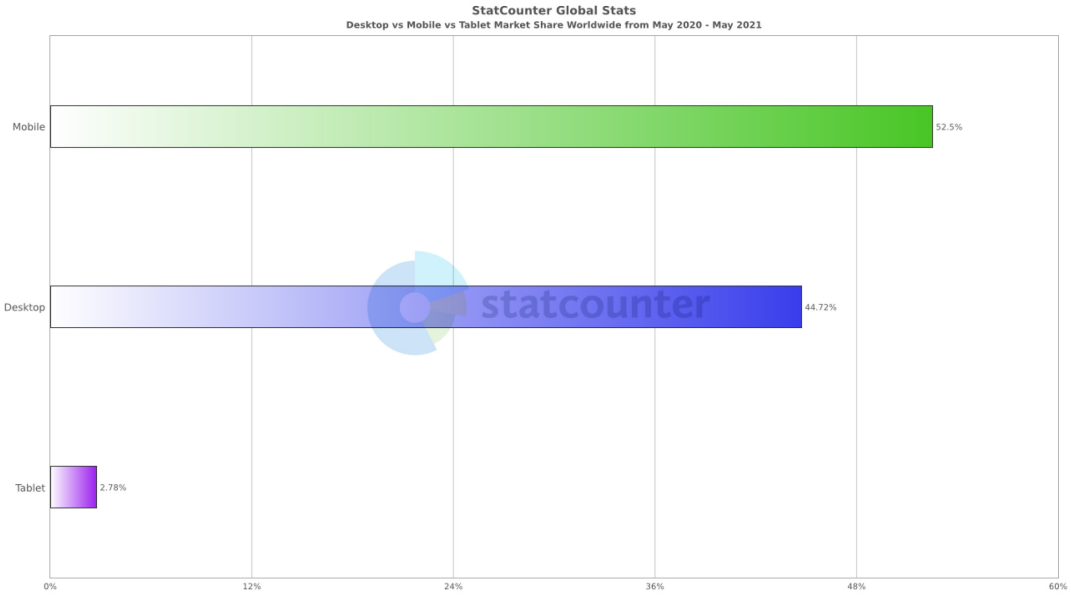
Vše je koncipováno okolo uživatelského pohodlí. Vzhledem k velkému množství webových stránek, které denně navštíví průměrný uživatel internetu, je důležité udělat web tak, aby plnil svůj účel aniž by bylo třeba dlouze zkoumat jak např. funguje ovládání galerie či kde se nachází hlavní menu. (Ho Tran, 2019)

Design ale nekončí v grafickém editoru. Je ho potřeba správně převést do kódu. Obě fáze návrhu, tedy výroba vizuálního podkladu v grafickém editoru i

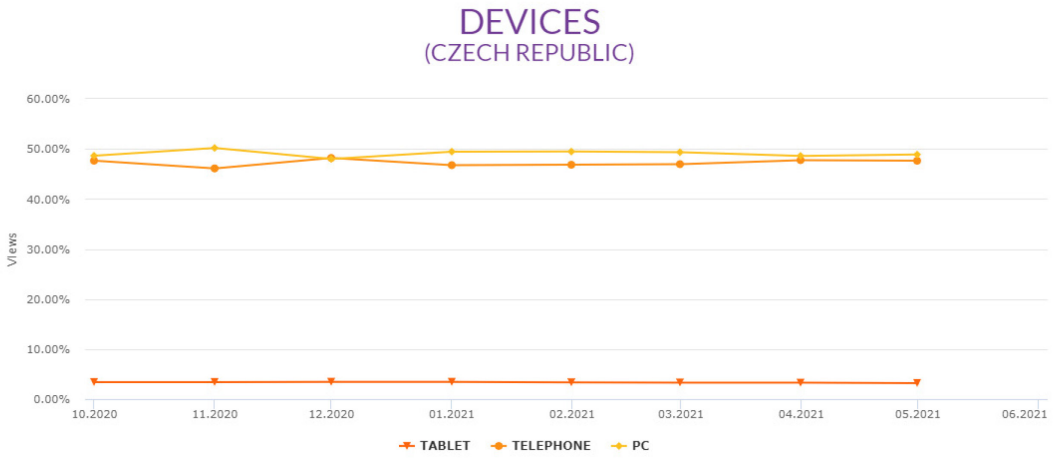
následný převod do HTML a CSS, spolu úzce souvisí. Proto je v grafickém editoru nutné myslet na kód a v textovém editoru na grafiku.

## 2.1 Responzivní design

Responzivita je vlastnost webu měnit svůj layout tak, aby byl plně funkční a vypadal dobře na různých obrazovkách, zařízeních a rozlišeních. Vzhledem k tomu, že většina návštěv webů v dnešní době je z mobilních zařízení, je responzivní design nutností, nikoliv bonusem.




*Obrázek 4 Celosvětový tržní podíl desktopů, mobilů a tabletů v období květen 2020–květen 2021 podle agentury StatCounter (Zdroj: StatCounter)*




*Obrázek 5 Tržní podíl desktopů, mobilů a tabletů v České republice v období říjen 2020–květen 2021 podle agentury Gemius s.r.o. (Zdroj: Gemius)*

Často se na responzivitu hledí pouze jako na adaptaci webu pro smartphony. To je však omyl, protože do ní spadá i ladění rozvržení pro širokoúhlé obrazovky s velkým rozlišením a displeje s nestandardním poměrem stran. Rozmanitost přístrojů, ze kterých lze přistupovat k webovému obsahu, pochopitelně komplikuje celý proces designu a vývoje. Práce navíc se ale v tomto případě vyplatí, protože s sebou nese velké množství výhod.

Stránky s responzivním rozložením jsou například upřednostňovány při zobrazení výsledků v Googlu, jelikož je jejich crawler[8] indexuje prioritně. (Google, 2016) Jak již bylo uvedeno, velká část online provozu pochází z tabletů a smartphonů, takže webstránka, která se na těchto zařízeních ovládá pohodlně, má větší šanci, že se na ní budou návštěvníci vracet. Toto je obzvlášť důležité pro komerční stránky typu e-shop a další online byznys.

### 2.1.1 Media Queries

Samotná implementace responzivního designu probíhá v CSS pomocí tzv. Media Queries. Prvně se definují podmínky, za kterých bude bude kód proveden a následně se zapisují samotná pravidla pro dané rozmezí. Syntaxe vypadá takto:

```
@media screen and (max-width: 768px) {
  .container {
    flex-direction: column;
  }
}
```

Pomocí Media Queries se upravuje stránka pro obrazovky s různými rozlišeními a tvoří se layout pro tisk. Podmínky lze i kombinovat a řetězit následujícím stylem:

```
@media screen and (max-width: 20em) and (orientation: landscape) { ... }
```

[8] Crawler je označení pro robota, který automaticky hledá a indexuje nové stránky.

Lze je zapisovat do stejného souboru jako zbylé CSS, nebo do zvláštního souboru, který bude vyhrazený pouze pro Media Queries. Výhodou druhé metody je přehlednost a potenciálně i lepší udržitelnost. Přísná pravidla ale neexistují.

### 2.1.2 Princip Mobile first

„Mobile First je způsob návrhu uživatelského rozhraní, který z pohledu důležitosti staví mobilní zařízení minimálně na úroveň tradičních počítačů s velkými obrazovkami.“ (Michálek, 2015) To v praxi znamená, že se jak samotný návrh tak i následné kódování dělá nejdříve pro mobily. Až když je vše vyladěné pro mobilní zařízení se řeší desktopové varianty. To však neznamená, že se kvalita snižuje se stoupajícím rozlišením.

Tento přístup je stále populárnější a oblíbenější ve světě webdesignu. Naprostá většina současných frontendových frameworků a knihoven již mobile first metodu adoptovala.

Typickými prvky (obecně responzivního designu, nikoliv pouze mobile first) jsou např. hamburger menu (navigace webu se rozbalí po kliknutí na ikonku, která vzdáleně připomíná hamburger), zarovnání elementů do jednoho sloupce, tedy roztáhnout stránku do výšky (single column layout) a velká tlačítka a input pole pro pohodlné mačkání na dotykových obrazovkách.

## 2.2 Webová přístupnost

Web byl stvořen pro všechny, nezávisle na jejich věku, úrovně vzdělání, fyzických či mentálních schopností či zdravotním stavu. Aby však mohla webová aplikace sloužit dobře i lidem se zdravotními postiženími, musí splňovat určitá kritéria a musí být postavena tak, aby nepředstavovala pro takové jedince překážku. Webovou přístupnost lze chápat jako sadu pravidel a doporučení pro vybudování inkluzivních webových stránek, které mohou plnohodnotně využívat i uživatelé s poruchami zraku, sluchu či motoriky. Pokud se postupuje při tvorbě webu od samého začátku s myšlenkou udělat web přístupný co nejvíce lidem, pak je dosažení uspokojivé míry přístupnosti relativně snadné. Stačí se držet základních doporučení.

Dobře čitelné písmo je nesmírně podstatné a ocení ho i lidé, kteří sice netrpí vadou zraku, ale nechtějí namáhat své oči. Lze tomu dosáhnout dostatečným kontrastem a větší velikostí. I při zvětšení textu na 200% byl měl být

návštěvník schopen se dostat na všechny části webu a k veškerému obsahu, nemělo by to tedy nijak negativně ovlivnit jeho zkušenost. (Thomson, 2020)

Další klíčovou složkou je kontrast. Ten se měří nejčastěji pomocí tzv. kontrastního poměru a měl by být alespoň 4,5:1 pro normální text, 3:1 pro velký text (18px a větší) a 3:1 pro běžné prvky uživatelského rozhraní aby splňoval požadavky WCAG 2.0 od W3C.[9]

Inkluzivita webu se řeší i na úrovni kódu a to pomocí sémantických značek, kterými disponuje HTML5. Je jednoduché vše zabalit do generických div tagů a „dál nad tím nepřemýšlet“, ale pro člověka, který k prohlížení daného webu používá například screen reader[10], bude nesmírně obtížné se orientovat a navigovat. Proto je dobré využívat sémantických tagů jako jsou section, aside, header, footer, nav a další.

Webová přístupnost je dokonce povinná pro stránky orgánů veřejné správy a státních institucí v Evropské Unii. Toto nařízení upravuje směrnice o přístupnosti webových stránek a mobilních aplikací subjektů veřejného sektoru z roku 2016, kterou vydaly Evropský parlament spolu s Radou EU.[11]

## 2.3 Layout

Při tvorbě designu webu se nejčastěji začíná právě rozložením jednotlivých prvků – kde a co na stránce bude (= layout). Dnes se zpravidla používá již standardní layout, který má v horní části hlavičku (header), pak hlavní obsah (main content) a úplně dole patičku (footer).

Do hlavičky se dává logo, hlavní navigace a případně i hledací pole či další důležité odkazy, které nejsou součástí hlavní navigace.

Následuje hlavní část webové stránky kde se nachází obsah. Vzhledem k množství a rozmanitosti webů zde není možné vymezit moc pravidel. Často tady bývá uložena i vedlejší / sekundární navigace, ale v zásadě se zde nejčastěji setkáme s textem, obrázky a multimediálními prvky, kterými nám webstránka komunikuje svůj účel.

[9] Úrovní webové přístupností je více (Level AA, Level AAA, ...) Viz kompletní znění od World Wide Web Consortium (W3C) https://www.w3.org/TR/WCAG20/
[10] Čtečka je zařízení které převádí obsah na Braillovo písmo nebo na zvukovou přehrávku.
[11] Viz https://eur-lex.europa.eu/legal-content/CS/TXT/HTML/?uri=CELEX:32016L2102 pro kompletní znění.

Ve spodní části stránky se nachází footer, neboli patička, kde je zpravidla prostor na právní informace, rychlé odkazy, kontaktní údaje, odkazy na sociální sítě a další.

Vzhledem k situaci na trhu, kde je čím dál tím více návštěv z mobilních zařízení, se doporučuje začít nejprve s návrhem rozvržení pro smartphony, tedy od nejmenšího viewportu[12] (aplikace principu mobile first), přes tablety a až nakonec se vypracovat k desktopové podobě.

## 2.4 Typografie

Typografie je umělecko-technický obor, který se zabývá písmem a textem. V kontextu webového vývoje a designu je důležité zvolit především dobře čitelné a esteticky smysluplné fonty.

Dobrým pravidlem je použít bezpatkové písmo (Roboto, Helvetica, Poppins, …) pro nadpisy a další velké texty a patkové písmo (Roboto Slab, Merriweather, PT Serif, …) pro odstavce, paragrafy a další dlouhé části souvislého textu. Ideální je použít 2 až 3 rodiny písem.[13]

Velikostí písem lze velmi efektivně zvýraznit vizuální hierarchii. Nadpisy, slogany a důležité informace by měly být výrazně větší než ostatní texty na stránce. Důležitost lze vyjádřit i větší tloušťkou fontu (font-weight), která by pro zmíněné případy mohla být i 700 a výš.

U dlouhých odstavců se obecně doporučuje nastavit výšku řádku (line-height v CSS) cca 1,4x větší než je základní velikost písma v daném bloku kvůli lepší čitelnosti. (Elansary, 2020)

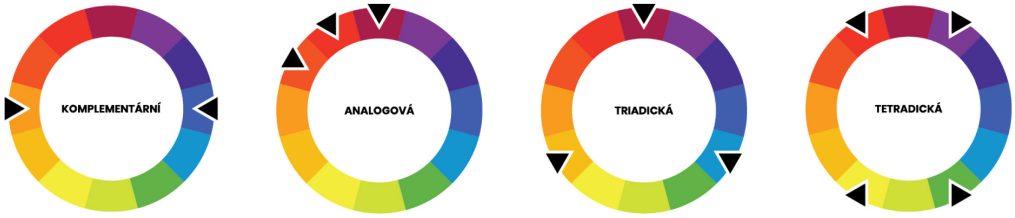
## 2.5 Barvy

Barevné schéma moderních webů se tvoří na základě analýzy, ze které se vyvodí cíl stránky a kdo je cílová skupina (kdo bude nejčastěji web navštěvovat). Každá barva na nás působí jinak a vyvolává v nás jiné emoce, současně na nás působí i podvědomě. Stojí za tím kombinace fyziky a psychologie, ale pro účely

[12] Viewport je označení pro momentálně zobrazenou část stránky v prohlížeči.
[13] Rodina písma (anglicky Typeface) obsahuje více variant fontu (různé tloušťky, styly a další). Příklad: Rodina písma Raleway zahrnuje fonty Raleway Regular, Raleway Bold, Raleway Black Italic a další.

základního pochopení stačí vědět, že je základem teorie barev, se kterou přišel na začátku 18.stol. Sir Isaac Newton.

Samotné barevné schéma se pak tvoří různými technikami na barevném kruhu. Nejčastěji používané jsou metody komplementární (2 barvy na opačných koncích), analogová (3 sousedící barvy), triadická (3 barvy, které jsou od sebe stejně vzdálené), tetradická (4 barvy, které jsou od sebe stejně vzdálené) a monochromatická (odstíny jedné barvy).




*Obrázek 6 Nejpoužívanější metody kombinace barev na barevném kruhu*

Základní asociace a významy jednotlivých základních barev:

- Žlutá: optimismus, štěstí, teplo, vřelost
- Oranžová: přátelská, veselá, energická
- Červená: vzrušující, vyžaduje pozornost, nebojácnost
- Fialová: kreativní, moudrost, nápaditá
- Modrá: důvěra, věrnost, bezpečí, pokrok
- Zelená: zdraví, příroda, čistota, růst
- Černá: elegance, síla, luxus, exkluzivita

Existuje spousta online nástrojů pomocí kterých lze vytvořit barevné schéma na web jako např. Adobe Color Wheel nebo Paletton. Další variantou je použít předpřipravené schéma.

## 2.6 Vektorová grafika, SVG

V minulosti se na grafické elementy s průhledným pozadím (ikonky, oddělovače, jednoduché tvary) používaly spíše formáty PNG a GIF, které umožňují práci s alfa kanálem. Ovšem dnešní standard je pokud možno se tomu vyhnout a použít místo toho vektorovou grafiku. Oproti bitmapové grafice totiž

nedochází u vektorů ke ztrátě kvality při škálování. V ideálním případě budou jedinými rastrovými prvky na stránce fotografie a videa. Ikonky, logo, ilustrace a jiné designové prvky, které lze vytvořit vektorem, mít ve vektorovém formátu.

Tím nejrozšířenějším je formát SVG (Scalable Vector Graphics). Je to značkovací jazyk, kterým popisujeme vektorová data, a formát souboru (koncovka .svg). První verze vyšla v roce 2001, ta nejnovější v roce 2011 (verze 1.1). SVG je velice rozšířené a oblíbené, mimo jiné, jelikož se s ním pracuje velmi snadno. Samotný kód lze přímo psát do HTML, kde je možné rovnou měnit i vzhled a styly. Pokročilejší úpravy jsou pak možné i pomocí CSS. Formát je nezávislý na platformě, podporují ho všechny moderní prohlížeče, velikost souborů je malá a může sobě obsahovat text a dokonce i bitmapový obrázek (ten však při zvětšování ztratí kvalitu).

## 2.7 Cross-browser compatibility

Responzivní design se stará o to, aby webovky vypadaly dobře na různých obrazovkách / zařízeních. Cross-browser compatibility (kompatibilita mezi prohlížeče) je snaha docílit toho, aby web vypadal a fungoval stejně, nebo jen s minimálními rozdíly, v různých prohlížečích. Je to další vrstva podpory, která má zaručit to, že bude webová stránka přístupná a pohodlná při práci co největšímu počtu uživatelů.

Každý internetový prohlížeč vykresluje stránky trochu jiným způsobem. Jsou dva hlavní důvody proč tomu tak je. Ten první z nich je render engine (vykreslovací jádro), které slouží k vykreslení / vizualizaci HTML a CSS souborů, obrázků a obecně všech prvků na stránce. Apple používá WebKit pro své prohlížeče (Safari, včetně mobilní iOS verze), všechny Chromium-based prohlížeče (Google Chrome, Brave, Vivaldi a další) používají Blink a Mozilla používá ve svém Firefoxu Gecko. Toto zdaleka není kompletní seznam, ale obsahuje všechna vykreslovací jádra těch nejpoužívanějších prohlížečů.

Druhý, a mnohem podstatnější, důvod proč existují odlišnosti v různých prohlížečích, i přesto, že vykreslují stejný kód, jsou built-in (vestavěné) styly, které každý z nich aplikuje na každé stránce. V Inspektoru (vývojářské nástroje

prohlížeče) jsou označeny jako „user agent stylesheet“ a typicky obsahují následující parametry a vlastnosti[14]:

- Velikost písma (nadpisy 1. až 6. úrovně, paragrafy)
- Padding a margin pro většinu objektů
- Styly pro input pole
- Vlastnost display pro většinu objektů
- Styly pro tlačítka
- Barvy různých textových objektů

Chceme-li výchozí hodnoty přepsat a využít čistě vlastní styly, je potřeba použít nástroj, kterému se říká CSS reset. Takových nástrojů je hodně (nejpoužívanější dnes je Normalize.css) je možné je do projektu jednoduše zakomponovat tak, že je do hlavičky uvedeme jako stylesheet link. Koneckonců se jedná pouze o dodatečné CSS, které resetuje výchozí styly prohlížeče.

Pro úplnost je nutno dodat, že Cross-browser compatibility se netýká pouze vizuální stránky webu nýbrž také i schopnosti korektně interpretovat skripty, zobrazovat různé soubory a multimediální elementy a další.

### 2.7.1 Vendor prefix

V případě, že chceme na stránce využít nějaké experimentální CSS vlastnosti (nebo JavaScript API), je potřeba k nim přidat tzv. vendor prefix. Jedná se o předponu, specifickou pro každý renderovací engine, která de facto funguje jako upozornění, že se jedná o nestabilní funkci a tudíž by neměla být užita v produkčním prostředí. Realita je však taková, že mnoho vývojářů tyto nestandardizované a pořádně neotestované možnosti zahrnuje do svých webových projektů i přesto, že hrozí, že daná věc nemusí na straně klienta dobře fungovat. (Mozilla, 2021) Vzhledem k situaci na trhu je dnes dostačující používat vendor prefixy pro WebKit (-webkit-) a pro Mozillu (-moz-). Chceme-li zahrnout i

[14] Viz https://chromium.googlesource.com/chromium/blink/+/refs/heads/main/Source/core/css/html.css (Chromium–based prohlížeče) https://searchfox.org/mozilla-central/source/layout/style/res/html.css (Mozilla Firefox), https://trac.webkit.org/browser/trunk/Source/WebCore/css/html.css (WebKit) pro kompletní znění

podporu pro Microsoft Edge či Internet Explorer, musíme použít jejich prefix ms. Zápis pak může vypadat např. takto:

```
-webkit-transition: all 500ms ease;
-moz-transition: all 500ms ease;
transition: all 500ms ease;
```

## 2.8 Relativní jednotky

Mezi doporučení a dobré praktiky pro moderní weby patří také užití relativních jednotek. Oproti absolutním jednotkám, které mají vždy pevně daný rozměr, se jejich velikost mění v závislosti na něčem jiném, nejčastěji na viewportu, rozlišení či předem definovaných hodnot.

Mezi absolutní jednotky patří milimetry (mm), centimetry (cm) nebo body (pt). Tyto jednotky jsou přesné a užitečné ve fyzickém světě, ale ve světě digitálním, kde se vše musí dodatečně přepočítávat a aproximovat, moc dobře nefungují. Zvláštním druhem absolutní jednotky je pixel. Ten se totiž liší zařízení od zařízení. U high-DPI monitorů je obrazový bod fyzicky mnohem menší než u „běžných“ monitorů.

Zástupci relativních jednotek jsou například procenta (%), em, rem, viewport width (vw), viewport height (vh). Pro text jsou určeny hlavně em a rem. Em se odkazuje na velikost písma (font-size) u rodičovského elementu. Příklad: Máme paragraf s velikostí písma 16px a uvnitř je span. Zadáme-li velikost tohoto span-u na 2em, bude velikost jeho fontu 32px. Rem funguje velice podobně, ale odkazuje se na velikost písma nastavená v root elementu (standardně 16px). Pomocí jednotky vycházející z viewportu (vw a vh) lze přesně nastavit jaká část (kolik procent) viewportu má daný prvek zabírat. Je samozřejmé, že půlka vertikálního a horizontálního rozměru okna internetového prohlížeče na desktopovém monitoru bude značně odlišná od té na mobilním zařízení.

Flexibilní jednotka fr (frakce) se moc často nepoužívá a vyskytuje se zatím v rámci CSS Grid sytému, kde slouží k určení velikosti, kterou dopočítá na základě zbývajícího, volného místa.

# 3 Současné možnosti využití frontendových knihoven pro vývoj webových stránek

Hned na začátku je podstatné si definovat dva základní pojmy, které se budou v práci často objevovat a je tudíž podstatné si vymezit jejich význam:

- **Front end** (taktéž **frontend** nebo front-end): Část webové stránky, kterou vidí a se kterou interaguje uživatel. Jedná se prezentační vrstvu webu, která návštěvníkovi podává v nějaké grafické podobě data a informace. „To, co vidíme při návštěvě webu."

- **Framework** (česky **knihovna**) je sada hotových kusů / bloků kódu organizovaných do souborů, která slouží k rychlejší a jednodušší tvorbě aplikací či tvorbě aplikačních prototypů. Webové knihovny tedy slouží k vývoji vizuální části webových stránek a aplikací a zrychlují a usnadňují práci vývojářů (za předpokladu, že daný framework dobře znají, samozřejmě). Podrobněji budou konkrétní frameworky popsány v dalších kapitolách práce.

Pro přesnost je nutno dodat, že knihovna (anglicky library) a framework nejsou synonyma. Ačkoliv jsou si velice podobné, existují mezi nimi rozdíly. Ten hlavní spočívá v tom, že u knihovny rozhoduje o tom co, kde a jak bude samotný vývojář (kdy nějaké komponenty knihovny vyvolat, kde určité elementy použít). U frameworku je tomu obráceně. Framework má již nějakou pevně definovanou strukturu a požaduje od programátora doplnění vlastního kódu. Knihovny mají většinou poměrně úzké zaměření, frameworky mají značně širší oblast využití. (Wozniewicz, 2019) Pro účely této práce však budou tyto termíny často používány zaměnitelně, protože CSS knihovny jsou jistým druhem frontendových frameworků (spadají do této kategorie) a u mnohé z nich není možné pevně vymezit o jaký typ nástroje se přesně jedná.

Vytvořit krásný, funkční a přístupný web v dnešní době je často náročný úkol. Návrhy grafiků mohou být složité na realizaci a časově velmi náročné. Další

možnost je, že vývojář nemá vůbec žádný design a potřebuje vše udělat sám. Obzvlášť u menších projektů jako jsou např. stránky open source projektů, frontend a prezentační část nějaké aplikace, osobní stránky, jednoduché webové aplikace a další, kde jsou čas a energie věnovány spíše funkcionalitě, jsou potřeba pomocné nástroje, které usnadní tvorbu vizuální části. Nástroje, které se umí alespoň do určité míry postarat o design, responzivitu, přístupnost, rozdíly v renderování stránky různými prohlížeči a umí ušetřit kodérům čas.

Při tvorbě webových stránek se některé úkoly opakují neustále dokola. Vesměs v každém projektu se musí nastavit nějaké paddingy u input elementů, styly tlačítek, mezery v tabulkách, vynulovat výchozí hodnoty prohlížeče, nastavit lepší box-sizing (preferovaný je border-box) a další. Stává se z toho rutinní práce.

Řešením výše uvedených problémů jsou frontendové frameworky, konkrétně CSS knihovny a frameworky, mající za cíl pomoci v tvorbě uživatelských rozhraní webovek tak, že nabídnou sadu nástrojů, formou předpřipravených bloků kódu.

Online k dispozici je skutečně mnoho frontend knihoven. Některé jsou malé, jiné nabízí velké množství funkcí a komponent. Některé z nich se soustředí pouze na jednu specifickou funkcionalitu (tlačítka, typografie, ikonky, animace, …). Jiné jsou univerzální a poskytují komplexní sadu nástrojů pro tvorbu designu webstránky (především na ty je tato práce zaměřena).

Většina těch nejpoužívanějších ale sdílí jisté vlastnosti, které jsou pro moderní webový vývoj klíčové. Jsou to právě ty principy uvedené v druhé kapitole, které jsou dnes nedocenitelné a které i ty nejlepší z nich musí dodržovat.

Zajímavý je každoroční průzkum State of CSS, dotazující se frontend vývojářů na jejich názor a zkušenosti s různými technologiemi souvisejícími s CSS. V kapitole o CSS knihovnách se dotazují na spokojenost, zájem, používání a informovanost ohledně těch nejpopulárnějších z nich. Výsledky následně porovnávají s těmi z předchozího roku.

V kategorii spokojenosti vyhrál Tailwind CSS s 87 %, zlepšení o 6 % oproti roku 2019. Druhé místo drží PureCSS a na třetí pozici je Bulma s 61 %. Největší zájem jevili vývojáři v roce 2020 o Tailwind CSS, PureCSS a Semantic UI. Nejpoužívanější a nejznámější framework byl v roce 2019 i 2020 Bootstrap. Překvapivé je ale, že spokojenost s ním klesla pod 50 % a předchozí rok v průzkumu také nedopadl moc dobře (52% spokojenost). (State of CSS, 2020)


*Obrázek 7 Spokojenost vývojářů s jednotlivými frameworky v roce 2019 a v roce 2020 podle dat ze State of CSS (Zdroj: State of CSS)*

## 3.1 Knihovny zvolené pro tuto práci

U každé z vybraných knihoven jsou uvedeny základní vlastnosti a informace. Zvlášť u rozsáhlých sekcí, jako jsou nástroje na pozicování a tvorby layoutu nebo personalizace, je zde kladen důraz pouze na to nejpodstatnější. Všechny informace a detailnější popisy jsou vždy k dispozici v dokumentaci příslušné knihovny.

### 3.1.1 Kritéria volby

Z velkého množství dostupných všestranných frontend CSS knihoven jsou pro tuto práci zvoleny 3, které odpovídají určitým kritériím a splňují jisté podmínky.

V prvé řadě se musí jednat o populární a komunitou oblíbený projekt, který vývojáři a designéři využívají v roce 2021. Tímto se profiltruje velký seznam, jehož podstatná část je tvořena malými knihovnami, které ačkoliv nemusí být o nic horší než velcí hráči v oboru, nemají zatím takové zastoupení na trhu a tak početnou komunitu (a tudíž i podporu). Kromě standardních metod zjištění popularity a oblíbenosti, jako jsou seznamy těch nejlepších a nejpoužívanějších

CSS knihoven pro rok 2020/2021 na webech, které se specializují na webový vývoj, front end a design[15], se dá spolehlivě vycházet i z dat z GitHubu. Oficiální repozitář musí mít alespoň 20 000 hvězdiček a musí ho využívat alespoň několik tisíc lidí (sekce „Used by“). Dále musí mít podle dat ze serveru Openbase minimálně 10 000 stažení týdně konzistentně za poslední půlrok a mít uživatelské hodnocení alespoň 4,5 z 5. Framework musí být registrován na stránce StackShare a mít tam alespoň 50 sledovatelů a figurovat jako součást stacku alespoň jedné firmy. V neposlední řadě je třeba, aby byla knihovna v seznamu detekovatelných technologií firmy Wappalyzer. Pravidla tohoto kritéria nebyla zvolena na základě přesných statistik či výpočetních modelů, ale na základě pozorování průměrných čísel a hodnot v uvedených kategorií pro projekty, které jsou ve sféře frontend designu všeobecně známé. Použitím tolika různých zdrojů a parametrů je jistota, že je toto kritérium poctivě splněno.

Knihovna musí nabízet nástroje pro tvorbu plně responzivního rozvržení stránky (nemusí to být nutně mobile first), alespoň základní styly a komponenty pro moderní web (tlačítka, formuláře, text) a sjednocení stylů pro všechny běžné prohlížeče. Tyto podmínky už z definice splňuje většina univerzálních frameworků, které jsou oblastí zájmů této práce, ale jsou zde uvedeny pro úplnost.

Další bod, který je nutné, aby framework splnil, je, že musí být v aktivním vývoji. Z oficiálních stránek nebo repozitářů musí být znát, že se vývojáři aktivně projektu věnují a že jsou plány i pro jeho budoucí rozvoj. Rozumné pravidlo je, že poslední stabilní verze (release) nesmí být starší než 6 měsíců.

Čtvrtým kritériem je, že projekt musí být open source. Kód by tedy měl být volně dostupný ke stažení a licence nesmí nijak omezovat použití knihovny (ani pro komerční využití).

Pátým a poslední kritériem je, že se vybrané knihovny musí od sebe nějak zásadně lišit. Ať už by to bylo jejich cílem, způsobem použití, filozofií a přístupem k problematice, nebo hlavním zaměřením.

[15] Viz https://dev.to/theme_selection/best-css-frameworks-in-2020-1jjh nebo https://www.lambdatest.com/blog/best-css-frameworks-2021/

### 3.1.2 Bootstrap

#### Obecná charakteristika

Bootstrap je nejznámější a nejpoužívanější frontendový UI framework na světě. Vznikl v roce 2010, ve společnosti Twitter, a momentálně je nejnovější verze 5. Nabízí obrovské množství předpřipravených komponent a utilit, běžně se vyskytující na webu (tlačítka, formuláře, karusely, dropdown nabídky, tabulky, karty, modaly a mnoho dalších), pokročilou typografii a rozšiřitelné barevné schéma. Jako jediný framework z této práce obsahuje JavaScript pluginy, které z něho dělají víc než CSS knihovnu.

Popularitu si zasloužil především flexibilitou a snadným používáním. Pracovat s Bootstrapem mohou jak zkušení kodéři tak i lidé, kteří jsou mírně pokročilí v HTML a CSS. Flexibilita spočívá v tom, že lze do aplikace zakomponovat pouze ty části frameworku, které jsou skutečně potřeba. Pokud například JavaScriptí zásuvné moduly nejsou potřeba tak je možné je úplně vypustit. Lze importovat i pouze určité části z CSS. Chceme-li využít pouze jejich grid, můžeme. Jeho flexibilita spočívá i v tom, že jej lze snadno zakomponovat i do jiných frameworků. Bootstrap se často kombinuje s komplexními frontendovými javascriptími frameworky jako Angular (ng-bootstrap), React (React Bootstrap) a Vue (BootstrapVue), kde jeho role je postarat se o design. Je velmi snadné ho ale přidat prakticky k jakémukoliv webu či programu, postaveném na webových technologiích (Electron) jelikož nabízí více způsobů instalace. Klasicky pomocí CDN, správcem balíčků npm, nebo přímo stažením zdrojových a komprimovaných souborů z oficiálního GitHub repozitáře. Navíc dává vývojářům možnost si prakticky vše upravit dle vlastních potřeb. Ať už to budou rozměry písem, mezery mezi elementy, úpravy responzivity, přidání vlastních barev nebo třeba nastavení JavaScript komponent. To je ovšem možné pouze v případě, že pracujeme se zdrojovými soubory (ideálně instalace přes npm) a máme tak přístup k Sass souborům, kde se nachází jednotlivé proměnné. Při použití CDN je sice také možné hodně hodnot upravit ve zvláštním CSS, ale je to pracnější a více limitující metoda.

Framework je známý tím, že obsahuje silně nastylované elementy, z čehož plyne i jedna z jeho největších výhod (ale i nevýhod). Jelikož se na Bootstrapu podílí i mnoho designerů, vypadá vše velice profesionálně a úhledně. Mnoho

programátorů proto nechává výchozí vzhled. Na tom by za normálních okolností nebylo nic špatného, ale vzhledem k tomu, že ho používá více než 3,4 milionů webových stránek[16] (skutečný počet může být i stonásobný) tak šance, že se budete často setkávat s defaultním nastavením není úplně malá. Stránky důsledkem toho sdílí prakticky stejný vzhled.

**Nástroje pro layout a pozicování**

Bootstrap používá dnes již ustálený 12sloupcový grid systém (grid-based layout, rozložení do mřížky), který je plně responzivní a dokonce mobile first. Celý layout systém je založen na CSS Flexbox a jednotlivá pravidla jsou psaná v Sassu.

Základní stavební kámen je tzv. kontejner (container), který upravuje pozici, zalomení a chování při změně viewportů, padding a případně i vycentrování obsahu, který se uvnitř něj nachází. Ten může mít buď přednastavenou maximální šířku, měnící se pro každý viewport (výchozí nastavení), nebo 100% šířku nezávisle na velikosti okna prohlížeče (container-fluid), nebo 100% šířku ale pouze do určitého breakpointu.[17]

Breakpointů je celkem 6: nejmenší pro rozlišení do 576px (výchozí stav, nemá infix) a největší (xxl) pro šířku 1400px a vyšší. Ostatní 4 mezistupně mají infixy sm, md, lg, a xl. Všechny hodnoty jsou zvoleny podle nejčastějších rozlišení viewportů smartphonů, tabletů, notebooků a high-DPI monitorů a jsou dělitelné dvanácti. Je možné je upravit dle vlastních potřeb v souboru _variables.scss. Jsou k dispozici navíc i Media Queries, ale z důvodů nepřehlednosti a potenciálních konfliktů s nastavenými breakpointy se to nedoporučuje.

[16] Viz data Wappalyzer-u: https://www.wappalyzer.com/technologies/ui-frameworks/
[17] Breakpoint (volně přeložené do češtiny jako „bod zlomu“, ale někdy i zarážka) je hraniční bod určité šířky, kterou definujeme v CSS pro vymezení responzivních úrovní.

*Tabulka 1 Breakpointy a responzivní systém Bootstrapu*

| Breakpoint | Třídní infix | Rozměry |
|---|---|---|
| X-Small | *Žádný* | <576px |
| Small | sm | ≥576px |
| Medium | md | ≥768px |
| Large | lg | ≥992px |
| Extra large | xl | ≥1200px |
| Extra extra large | xxl | ≥1400px |

Třída .row vytváří nový řádek a třída .col slouží k přidání sloupce. Na řádku můžeme mít libovolný počet sloupců (avšak maximálně 12) a každý z nich může zabírat libovolnou část dělitelnou 12. Grid systém Bootstrapu umožňuje nastavit šířku sloupců i v závislosti na breakpointů. Pomocí speciálních tříd můžeme určit jak se bude daný element chovat při různých rozlišeních.

```
Element s pevnou šířkou: Obecná formulace zápisu:
{col}-{šířka 1 až 12}

Příklad:
<div class="col-6">Tento element má vždy 50% šířku.</div>

---

Element s měnící se šířkou podle breakpointu: Obecná formulace zápisu:
{col}-{šířka 1 až 12} {col}-{breakpoint}-{šířka 1 až 12}

Příklad:
<div class="col-6 col-md-4">50% šířka na mobil a 33% od md dál.</div>
```

Všechny možnosti Flexboxu jsou samozřejmě k dispozici pomocí speciálních tříd, jejichž jména se shodují s názvy vlastností. Třída d-flex přidá kontejneru vlastnost display: flex. Vlastnost justify-content lze přidat třídou .justify-content, align-items: baseline třídou .align-items-baseline atd. Stejně jako u sloupců i zde můžeme vše aplikovat pouze na určité viewporty.

```
Obecná formulace zápisu:
{vlastnost}-{breakpoint}-{hodnota}

Příklad:
```

```
.align-self-xl-center
```

Přidávání margin a padding funguje úplně stejně, pomocí tříd. Boostrap nabízí 6 různých hodnot pro obě vlastnosti (0 až 5) přičemž každá úroveň se odvíjí od proměnné $spacer v souboru s utilitami. Výchozí hodnota $spacer je 1 rem. Hodnota 0 maže padding / margin, hodnota 1 má vzorec $spacer * .25, hodnota 2 $spacer * .5, hodnota 3 je výchozí atd. Třídy jsou pojmenovány logicky, pracuje se s nimi velmi snadno a po několika využitích již není zapotřebí hledět do dokumentace.

```
Obecná formulace zápisu:
{vlastnost}-{strany}-{velikost}

Příklady:
pb-4 (padding-bottom: 4)
mr-0 (margin-right: 0)
px-5 (padding vlevo i vpravo, osa X)
my-1 (margin nahoře i dole, osa Y)
```

Pozicování pro „přilepení“ elementů se také dělá pomocí tříd; .fixed-top, fixed-bottom a sticky-top.

To vlastně znamená, že můžeme tvořit i složité layouty jenom pomocí tříd, aniž bychom napsali jediný řádek CSS. Nemálo času ušetří i fakt, že v drtivé většině případů není potřeba psát Media Queries.

**Typografie**

Boostrap používá tzv. nativní font stack, což jsou výchozí systémová písma (odlišná pro každý operační systém), výchozí rozměr písma z root elementu prohlížeče a jednotky rem pro ostatní velikosti.

Pro nadpisy lze použít HTML značky h1 až h6, nebo třídy h1 až h6, pokud potřebujeme replikovat vzhled, ale sémanticky by to nedávalo smysl.

Framework má v sobě i speciální třídy .display-1 až .display-6, které vykreslují elegantnější písmo s menším font-weight. Třída .lead se používá u paragrafů, které jsou potřeba vizuálně odlišit od zbytku. Text je o něco větší a má světlejší barvu. Přeškrtnutý text, zvýrazněný text, citace, tučné písmo, kurzíva a mnohé další jsou zde samozřejmostí.

Vývojáři Bootstrapu si dali hodně záležet na typografii. Koneckonců se jedná o jednu z nejdůležitějších částí webové stránky. Vše je opět samozřejmě snadno upravitelné od základu v příslušném Sass souboru, pokud výchozí hodnoty nejsou uspokojivé.

**Barvy**

Není překvapující, že ani v této oblasti není výběr malý. Bootstrap disponuje bohatým barevným schématem, který by měl být pro většinu případů dostačující. V základu je ale obsaženo pouze několik málo konkrétních barev z důvodu ušetření velikosti. Modrá je .primary, zelená .success, červená .danger atd. Místo jmen barev je tedy využito sémantické názvosloví podle toho jakou roli budou s největší pravděpodobností plnit elementy, které budou takto zabarvené.

Zbylé barvy jsou k dispozici formou Sass proměnných. Tím pádem si můžeme vygenerovat CSS, které bude obsahovat pouze ty barvy, které skutečně využijeme ve svém projektu. Máme-li instalaci provedenou pomocí npm, můžeme jednoduše importovat potřebné barvy a jejich odstíny z rozšířené barevné palety.

```
@each $key, $value in $pinks {
 .bg-pink-#{$key}{
   background-color: $value !important;
 }

 .text-pink-#{$key}{
   color: $value !important;
 }
}
```

Pomocí tohoto jednoduchého zápisu a skriptu „npm run dev“ se nám do produkčního CSS souboru vygenerují předpřipravené třídy pro nastavení pozadí a barvy textu na růžovou a odstíny růžové. Samozřejmě, že pokud vytváříme webové stránky pro klienta, který již má vlastní barvy, tak není problém je tak dopsat.

**Běžné webové komponenty**

Tlačítka v různých barvách a provedeních, kartičky, rozbalovací nabídky, navigační lišty, formuláře a nesčetně mnoho dalších běžně užívaných komponent na webstránkách mohou být vytvořeny během několik minut. Jakožto tradiční

zástupce obsáhlých frameworků se Bootstrap snaží vývojářům nabídnout prakticky všechny běžné elementy, které by mohli potřebovat při tvorbě webů.

Jako jeden z mála CSS frameworků zahrnuje i JavaScript (nově od verze 5 bez nutnosti jQuery).[18] To umožňuje tvorbu modalů, již zmíněných rozbalovacích komponent, upozornění, nápovědy, vyskakovací upozornění, scrollspy[19] a další. Naprogramovat hamburger menu pro mobilní verzi je skoro triviální. Stačí přidat správné třídy a vlastnosti k elementům, které do toho patří a Bootstrap se postará o zbytek. Není potřeba psát i jediný řádek JavaScriptu. Dostupná je dokonce i validace formulářů, která ale se zatím stále doporučuje dělat spíše na straně serveru, jelikož styly a zpětná vazba / nápověda nejsou u asistenčních technologií podporovány.

Relativně nově nabízí Bootstrap i vlastní ikonky, které je ale možné využít i tam, kde Bootstrap nainstalovaný není. Kolekce se rychle rozrůstá, jsou všechny v SVG formátu a je možné je zakomponovat do designu mnoha různými způsoby (i jako web font). (Dokumentace Bootstrap, 2021)

**Dokumentace**

Dokumentace frameworku je poctivě zpracovaná a obsahuje veškeré potřebné informace pro pochopení toho jak funguje a jak ho efektivně používat. Všechna témata jsou roztříděna do patřičných kategorií a v horní částí je i hledací pole s klávesovou zkratkou pro ještě rychlejší vyhledávání. Obsahuje informace o projektu (kdy a proč vznikl), týmu, který ho vyvíjí, historii a vizí do budoucna. Možnosti použití a instalace jsou jednoduše popsané a srozumitelné. Všechny běžné komponenty jsou vysvětleny pomocí příkladu a pod ním je samotným kód, který je možno zkopírovat jedním kliknutím.

Na oficiálních stránkách je k dispozici volně ke stažení mnoho předpřipravených elementů (a hodně z nich v světlých i tmavých variantách) jako jsou tabulky cenových nabídek, fotogalerie nebo hero sekce, a také šablony celých stránek (blog, produktová stránka, osobní portfolio a další).

[18] jQuery je však možné použít spolu s Boostrapem i když reálně není moc důvodů proč. Jedinou externí JS knihovnou, kterou Bootstrap používá, je PopperJS. Ten je potřeba zahrnout pouze v případě, že potřebujeme využít tooltip možnosti (nápovědu).
[19] Scrollspy automaticky aktualizuje prvky v navigaci na základě toho, která sekce je právě zobrazena ve viewportu.

Dokumentace je celkově na výborné úrovni, ale vzhledem k počtu možností, které Bootstrap nabízí, je škoda, že chybí video návody.

### 3.1.3 Tailwind CSS

#### Obecná charakteristika

Tailwind CSS se staví k tvorbě uživatelských rozhraní naprosto odlišně od jiných populárních řešení v tomto odvětví jako jsou Bootstrap, Foundation, Materialize nebo Bulma. Všechny vyjmenované frameworky jsou totiž názorově vyhraněné co se týče designu a vzhledu a nabízí již hotové a vysoce nastylované komponenty. Mají vlastní designový jazyk. Tailwind nic takového nenabízí. Jde o tzv. utility-first framework, což znamená, že uživateli poskytuje nesmírně velké množství low-level tříd a utilit, pomocí kterých si kodér sám tvoří elementy a určuje jejich vzhled. S použitím Tailwind CSS nedochází k onomu „šablonovitému“ vzhledu, ke kterému často dochází při použití frameworků s výraznou grafickou nadstavbou (jako například výše zmíněné). Má dva hlavní účely: Poskytnout nástroje pro vytvoření jakéhokoliv designu a pracovat pokud možno pouze v HTML souboru, kaskádové styly používat minimálně. Podle autora frameworku Tailwind CSS, Adama Wathana, je rozptylující neustále přepínat mezi soubory. Tvrdí taktéž, že čím víc je nějaký element od základu nastylyzován, tím těžší je jeho opakované využití. S trochou nadsázky je možno říct, že se Tailwind snaží převést značnou část jazyka CSS do primitivních HTML tříd a vlastnosti každého elementu se určují právě pomocí těchto tříd.

Instalace a použití jsou v zásadě jednoduché a velice podobné Bootstrapu, ale hlavní a doporučená metoda je pomocí npm jako PostCSS[20] plugin. Možnost využití CDN tam sice existuje, ale autoři tuto metodu nedoporučují, protože se ztratí možnost personalizace, vlastních úprav, a smazání nevyužitých stylů. Celý, nekomprimovaný CSS soubor dosahuje velikosti téměř 3MB, na produkční web prakticky nepoužitelný. Proto má Tailwind funkci Purge, která smaže všechny nepoužité styly a nechá ve výsledném CSS pouze to, co je v projektu skutečně zahrnuto (z cca 3MB je ve finále většinou méně než 10kB).

[20] PostCSS je JavaScript nástroj pro kontrolu, úpravu, kompilaci a generování CSS. Oficiální web: https://postcss.org/

Jeden z velkých konceptů tohoto frameworku je vlastní konfigurace a nastavení. Po nainstalování přes npm lze vytvořit konfigurační soubor, ten nicméně povinný není.

```
npx tailwindcss init
```

Tento příkaz vytvoří soubor tailwind.config.js, ve kterém si vývojář může upravit různé části frameworku. Konfigurační soubor je rozdělen na 3 sekce: theme, variants a plugings. Do theme můžeme přidávat pravidla a úpravy pro písma, barvy, margin, padding, okraje, stíny a další prvky designu. Sekce variants slouží pro personalizaci například hover a focus stavů. V plugins se mohou psát vlastní styly, které chceme zahrnout do naší, personalizované, verze Tailwindu. Místo psaní přímo do CSS se doporučuje všechny vlastnosti navíc, které chceme opakovaně používat, zapisovat právě zde. Podstatné je po přidání jakéhokoliv nastavení spustit skript, generující nové CSS s nově přidanými funkcemi. Příklad konfiguračního souboru tailwind.config.js:

```
const colors = require('tailwindcss/colors')

module.exports = {
  purge: {
    enabled: true,
    content: ['./public/*.html']
  },
  darkMode: false, // or 'media' or 'class'
  theme: {
    colors: {
      transparent: 'transparent',
      current: 'currentColor',
      black: colors.black,
      white: colors.white,
      gray: colors.trueGray,
      blue: colors.blue,
    },
    extend: {},
  },
  variants: {
      borderColor: ['focus', 'hover'],
    extend: {},
  },
  plugins: [require('@tailwindcss/typography'),],
}
```

Tailwind CSS se doporučuje spíše pokročilým uživatelům, protože je potřeba dobře znát CSS, mít základní chápání webového designu (zvlášť v případě, že není k dispozici hotový design) a umět pracovat s npm a v Sassu alespoň na základní úrovni. V posledních letech jeho popularita stoupá.

**Nástroje pro layout a pozicování**

Tailwind CSS je plně responzivní, mobile first framework jehož layout nástroje jsou postavené na Flexboxu a na CSS Gridu. Obsah se často dává do kontejnerů, které mají šířku nastavenou podle předem nastavených breakpointů.

Breakpointů je celkem 5. Nejmenší má danou výchozí hodnotu maximální šířky 640px (sm) a největší 1536px (2xl). Mezistupně jsou md (768px), lg (1024px) a xl (1280px). Neuvedení žádného z nich automaticky znamená 100% šířka objektu. Všechny hodnoty je samozřejmě možné změnit v konfiguračním souboru.

*Tabulka 2 Breakpointy a responzivní systém Tailwind CSS*

| Třídní prefix | Rozměry |
|---|---|
| *Žádný* | <640px |
| sm | ≥640px |
| md | ≥768px |
| lg | ≥1024px |
| xl | ≥1280px |
| 2xl | ≥1536px |

Do tříd elementů se zapisují všechny jeho vlastnosti pro pozicování a zobrazení z kaskádových stylů formou zkrácených zápisů (utility tříd Tailwind CSS). V následujícím příkladě je header sekce, která bude mít vlastnosti display: flex, flex-wrap: wrap, justify-content: center a align-items: center a v případě, že bude viewport mít alespoň 768px (md) se hodnota vlastnosti justify-content změní na between.

```
<header class="container flex flex-wrap justify-center md:justify-between
items-center "> ... </header>
```

Tvorba rozložení je prakticky stejná jako při práci s běžným CSS, pouze je vše přeloženo do krátkých zápisů pomocí tříd. Není tu žádný vlastní grid systém, používá se klasický CSS Flexbox (se všemi jeho vlastnostmi, možnostmi a omezeními).

Přidání margin a padding funguje stejným způsobem. Jejich hodnotu pak zadáme číslem. Tailwind poskytuje mnohem větší množství výchozích hodnot než Bootstrap, ale v obou případech je možné je rozšířit o vlastní preference.

```
<div class="p-5">Padding 5 (1.25rem) do všech směrů</div>

<div class="my-3 lg:my-4">Margin 3 (0.75rem) a margin 4 (1 rem) od lg
breakpointu dál. Aplikuje margin-top a margin-bottom (osa Y)</div>
```

Framework obsahuje skoro 2 500 tříd pro margin (včetně záporného) a přes 2 000 tříd pro padding.

**Typografie**

Součástí základního balíčku Tailwind CSS jsou pouze některé běžné textové vlastnosti, samozřejmě opět formou utility tříd. Mezi ně patří tloušťka textu, velikost, a úprava barvy. Nadpisy nemají aplikované žádné styly. Více variant a stylů je možné přidat pomocí volitelného rozšíření Tailwind Typography, které obsahuje modifikace pro paragrafy, seznamy, nadpisy, citáty a jiné speciální formy textu.

Existuje též varianta tvorby vlastních komponent. Můžeme si v nastavení frameworku vytvořit vzory pro všechny výše zmíněné případy. Tailwind je velice flexibilní a volba je ve výsledku na uživateli.

**Barvy**

Barvy jsou řešeny při základním použití vesměs stejně jako typografické prvky. K dispozici je menší sada barev, ze které můžeme čerpat při stylování textu či pozadí elementů. Je to opět z důvodů šetření rozměrů a optimalizace. Tailwind CSS ale obsahuje velkou barevnou paletu pečlivě zvolených barev a barevných

odstínů. Stačí naimportovat v konfiguračním souboru ty sady barev, které potřebujeme k projektu. Narozdíl od typografie zde není potřeba využívat plugin.

**Běžné webové komponenty**

Tailwind v sobě nemá žádné hotové webové komponenty. Je potřeba aby si vývojář vše vytvořil sám. Není na tom nic překvapujícího, bereme-li v potaz filozofii na které je celý projekt založen.

Oficiální komponenty k Taiwlindu ale existují. Jsou ovšem placené.

Balíček ikon od tvůrců Tailwindu se jmenuje Heroicons, je bezplatný a open source. Jedná se však o zvláštní projekt, který není tak úzce spjat se samotným frameworkem, jako je tomu u Bootstrapu. (Dokumentace Tailwind CSS, 2021)

**Dokumentace**

Dokumentace je velice pěkně a přehledně zpracovaná. Stejně jako Bootstrap je Tailwind CSS poměrně rozsáhlý projekt, který obsahuje mnoho nástrojů a široké možnosti úprav a personalizace. Od instalace, přes samotné užívání, až po rozšíření a nastavení. Vše je dobře vysvětleno na konkrétních příkladech u kterých je k dispozici i kód, který lze zkopírovat jedním kliknutím.

Součástí je i online textový editor s živým náhledem, kde je možné si zkusit prakticky všechny možnosti. K dokumentaci jsou natočené i oficiální video návody přímo od autora.

### 3.1.4 Pure

**Obecná charakteristika**

Pure, nebo PureCSS, je velmi malá a minimalistická CSS knihovna vyvíjena programátory z Yahoo. Je známá tím, že celá, ve zkomprimované formě, má pouze 3,7KB a přitom obsahuje základní nástroje a pomůcky na výrobu moderní webové stránky. Je taktéž modulární, stejně jako Bootstrap a Tailwind CSS. Skládá se z 6 částí: Base, Grids, Forms, Buttons, Tables, a Menus. Jediná povinná složka je Base. Ta musí být vždy přítomna, pracujeme-li s knihovnou. Všechny nabízené elementy mají aplikované základní styly, ale při práci s Pure se počítá s tím, že si kodéři vše doplní podle vlastních potřeb.

Base balíček obsahuje Normalize.css[21] a pár drobných úprav od tvořitelů Pure, které souvisí s výchozím chováním prohlížečů.

Instalaci je možné opět provést tradičními metodami a to pomocí CDN, npm, nebo manuálním stažením souborů a jejich následné vložení do projektu.

**Nástroje pro layout a pozicování**

Pure má svůj vlastní grid systém se kterým se pracuje velmi rychle a jednoduše. Základem jsou grid třída (pure-g) a jednotková třída / unit elementy (pure-u nebio pure-u-*). Všechny potomky objektu s třídou pure-g musí mít třídu pure-u (nebo pure-u-*). Obecně vzato je třída pure-g ekvivalent kontejneru v Bootstrapu či Tailwindu. Šířky jednotlivých unit elementů je vždy dána zlomkem.

```
<div class="pure-g">
    <div class="pure-u-1-2">Tento div zabírá půlku (50%).</div>
    <div class="pure-u-1-2">Tento div zabírá také půlku (50%).</div>
</div>

<div class="pure-g">
    <div class="pure-u-1-4">Tento div zabírá čtvrtinu (25%).</div>
    <div class="pure-u-1-4">Tento div zabírá čtvrtinu (25%).</div>
    <div class="pure-u-1-4">Tento div zabírá čtvrtinu (25%).</div>
    <div class="pure-u-1-4">Tento div zabírá čtvrtinu (25%).</div>
</div>
```

Rozložení je možné na části dělitelné 5 nebo 24 pro větší přesnost.

```
<div class="pure-g">
    <div class="pure-u-3-5">Tento div zabírá 3/5 šířky.</div>
    <div class="pure-u-2-5">Tento div zabírá 2/5 šířky.</div>
</div>

<div class="pure-u-3-4">Tento div zabírá 3/4 šířky.</div>
<div class="pure-u-18-24">Tento div zabírá také 3/4 šířky.</div>
```

Tento systém má ovšem jednu nevýhodu. Rozměry jsou fixní a vytváří tudíž neresponzivní layout. Pure nabízí i responzivní Grid systém, který funguje velmi podobně, ale ten není zahrnut v balíčku Grids. Je tedy zapotřebí jej zvlášť

[21] Normalize.css je nástroj pro reset výchozích stylů prohlížečů a sjednocení vzhledu některých elementů. Oficiální web: https://necolas.github.io/normalize.css/

přidat (například přes CDN). Pak je možné, stejně jako u předchozích knihoven, ovládat šířku a zobrazení pro jednotlivé viewporty.

*Tabulka 3 Breakpointy a responzivní systém Pure*

| Třídní infix | Třída | Rozměry |
|---|---|---|
| Žádný | .pure-u-* | platí vždy |
| sm | .pure-u-sm-* | ≥ 568px |
| md | .pure-u-md-* | ≥ 768px |
| lg | .pure-u-lg-* | ≥ 1024px |
| xl | .pure-u-xl-* | ≥ 1280px |

```
<div class="pure-g">
    <div class=""pure-u-1 pure-u-lg-1-3">Tento div má 100% šířku na malá
zařízení a od lg breakpointu má šířku 33%.</div>
    <div class=""pure-u-1 pure-u-lg-1-3">Tento div má 100% šířku na malá
zařízení a od lg breakpointu má šířku 33%.</div>
<div class=""pure-u-1 pure-u-lg-1-3">Tento div má 100% šířku na malá
zařízení a od lg breakpointu má šířku 33%.</div>
</div>
```

Tento systém je možné kombinovat s Flexboxem a CSS Gridem.

### Typografie

Knihovna Pure neobsahuje žádné typografické nástroje. Úpravy textu se řeší standardně v CSS.

### Barvy

PureCSS neobsahuje žádnou barevnou paletu ani nástroje pro práci s barvou.

### Běžné webové komponenty

Tlačítka, obsažena v části Buttons, mají pouze minimální stylové úpravy. Jsou určeny k tomu, aby je uživatel upravil dle vlastního vkusu a potřeby. Jedině button-primary má již od vývojářů přednastavený styl modré barvy a bílého text. (obdoba Bootstrapu). Jemný hover efekt je také aplikován.

Formuláře, v balení Forms, jsou ale nečekaně propracované v porovnání s ostatními komponenty. Třída pure-form vytváří inline formuláře a pure-form-stacked vykreslí políčka pod sebe. Padding, placeholder a label text jsou mnohem lepší než ve výchozím stavu a pro mnohé projekty je nejspíš nebude nutné ani modifikovat.

Tables se stará o jednoduchý a přehledný styl tabulek a Menus poskytuje rychlý způsob jak vytvořit navigace. Oba tyto nástroje jsou ale velmi omezené. Tabulky by mohly být dostačující i v základu, ale navigační panely jsou spíše chudé než-li jednoduché, jak tvrdí vývojáři knihovny. (Dokumentace Pure, 2021)

**Dokumentace**

Ani Pure není výjimkou v této oblasti. Dokumentace je jasná, přehledná a stručná, ale zároveň obsahuje všechny potřebné informace. Malá velikost této knihovny je pro ní v určitých ohledech velkou výhodou. Lze se jí poměrně rychle naučit a začít operovat se vším co nabízí (je ale pravda, že toho není moc).

Velké plus dokumentace jsou i hotové šablony, jejichž kód je snadno dostupný na GitHubu. Jsou mezi nimi i blog, emailový klient, fotogalerie, stránka s responzivním menu vlevo a další.

## 3.2 Srovnání vybraných knihoven

Všechny tři frameworky řeší stejný problém jiným způsobem. Bootstrap nabízí prakticky všechny běžné webové komponenty a typografické a barevné nástroje pro tvorbu webu. Má svůj ikonický styl, který je aplikován na všech elementech. V případě, že nemáme k dispozici grafický návrh a chceme vytvořit moderní, plně responzivní web, pak je Bootstrap tou nejlepší variantou. Jelikož je vše předpřipravené, není potřeba plýtvat čas vymýšlením a tvořením vzhledu. V takových situací je jeho silnou stránkou i rychlost tvorby stránek. Nevýhodou Bootstrapu je, že vzhledem k výraznému designovému jazyku, který je patrný napříč všech jeho částí, vypadají weby velmi podobně. Elegantně, čistě, ale „šablonovitě“. Jeho popularita tomuto nežádoucímu efektu ještě napomáhá. I s Bootstrapem lze vytvořit originálně vypadající webovou stránku, ale pakliže je potřeba upravit styl každého z elementů, je lepší vybrat jinou knihovnu. Například právě Tailwind.

Tailwind CSS je určen převážně pokročilejším uživatelům, kteří dobře znají jednotlivé vlastnosti a možnosti jazyka CSS. Z nejdůležitější a nejpoužívanějších částí kaskádových stylů jsou udělané low-level třídy. Valná většina stylování tedy probíhá uvnitř HTML Tailwind nenabízí žádné hotové komponenty, moduly a šablony. Jeho filozofií je, že si vše dělá vývojář sám a pouze poskytuje potřebné nástroje. Cílem tohoto frameworku není umožnit tvorbu webové stránky co nejrychleji, ale umožnit v rozumném čase převést do kódu jakýkoliv design. Nejvhodnější je pro případ, že máme k dispozici grafický návrh (ať je jakkoliv složitý). Práce s ním je ale na začátku náročnější, v porovnáním s Bootstrapem.

Pure je už od základu hodně odlišný od předchozích dvou frameworků. Patří do tzv. minimalistických knihoven, které se vyznačují svojí malou velikostí a poněkud omezenou sadou funkcí. PureCSS pomáhá s monotónními úlohy jako jsou tvorba rozvržení, responzivita, formuláře a sjednocená stylů v prohlížeči (pomocí Normalize.css) a jako bonus přidává do projektu základní styly pro tlačítka, navigace a tabulky. Je vhodný pro menší projekty, nebo jako doplněk k většímu frameworku (i na oficiálních stránkách autoři uvádí jako příklad využití v kombinaci s Bootstrapem).

Tabulkové srovnání základních parametrů a možností je v příloze č.1. Ta byla vypracována na základě dostupných dat z oficiálních dokumentací a osobních zkušeností.

## 3.3 Další populární a často používané knihovny

Knihoven a frameworků, které splňují většinu kritérií (nebo dokonce všechny) definovaných v podkapitole „Kritéria volby“ je více a zaslouží si alespoň stručné představení. Zároveň u každé z nich bude uveden důvod (případně důvody) proč nebyly zvoleny v rámci vypracování praktické části této práce.

### 3.3.1 Bulma

Po Bootstrapu a Tailwindu je Bulma často uváděna jako třetí nejrozšířenější CSS framework. Jedná se o bezplatnou, open source, plně responzivní (mobile first) knihovnu založenou na Flexboxu, která nabízí předpřipravené frontendové komponenty a nástroje pro rychlou výrobu moderních webů. Cílovou skupinou jsou převážně začátečníci a lidé, kteří obecně nemají

moc zkušeností s podobnými technologiemi. Na svých oficiálních stránkách dokonce uvádí, že na použití této knihovny „Znalost CSS není nutná“.

Instalace je možná pomocí CDN nebo přes npm. Druhá varianta, stejně jako u všech ostatních, umožňuje úpravu proměnných jako jsou breakpointy, barvy, textové efekty a skoro všech ostatních stylů.

De facto všechno, včetně tvorby rozložení, volby barev, velikostí a efektů, je v Bulmě řešeno pomocí tříd (podobně jako u Tailwindu). Jména tříd začínají buď na „is“ nebo „has“. Chceme-li například vytvořit tmavé pozadí objektu, použijeme třídu „has-background-dark“. Vycentrování textu řeší třída „has-text-centered“. Objekt s třídou „is-one-third“ pak má šířku jedné třetiny svého rodiče. Vše je laděno tak, aby bylo co nejsrozumitelnější pro nováčky, kteří se nemusí ještě dobře vyznat v tvorbě frontend designu. (Dokumentace Bulma, 2021)

Ačkoliv splňuje Bulma skoro všechny stanovené požadavky, nebyla zvolena kvůli tomu, že se neodlišuje dostatečně od Bootstrapu (který z pochopitelných důvodů musel být nutně zahrnut v trojici detailněji zkoumaných). Bulma dělá vesměs to stejné co Bootstrap, ale nabízí značně méně nástrojů a je upravena pro jinou cílovou skupinu.

### 3.3.2 Materialize

Materialize je zajímavý framework založený na material designu. Jeho součástí je extenzivní barevná paleta a mnoho často používaných webových komponent (navbar, ikonky, tlačítka, karty). Díky JavaScriptu je možné bez dalších knihoven vytvářet modaly, parallax efekt, scrollspy a jiné. Vše laděné do již zmíněného Google material designu, zahrnující krásné a plynulé animace a konzistentní, profesionální vzhled na všech zařízeních.

Materialize poskytuje všechno co od moderního, univerzálního frontend frameworku očekáváme: Instalace pomocí CDN i npm, možnost vlastních úprav skoro všech hodnot, responzivní mobile first návrh, design pomocí HTML tříd a kvalitní dokumentace. (Materialize CSS, 2018)

Bohužel projekt již přes 3 roky nebyl aktualizován a o jeho budoucnosti je možné zatím pouze spekulovat.

### 3.3.3 Foundation

ZURB Foundation je typickým příkladem všestranného frontend frameworku, který se snaží poskytnout webdesignerům vše v jednom balení. První verze vyšla v roce 2011 a od té doby si vybudovala Foundation dobré jméno a reputaci ve frontend komunitě. Je často označována jako „nejpokročilejší frontend framework na světě“.

Obsahuje prakticky jakýkoliv často užívaný element na webu, pokročilé nástroje pro responzivní rozložení (XY grid systém), typografii, barevné schéma, JavaScript moduly, a integraci doplňků třetích stran. Z oficiálních stránek je možné si stáhnout nespočet šablon pro různé účely a má speciální verzi pro tvorbu e-mailových šablon.

Dokumentace je bohatá a velmi obsáhlá. To je skoro nutnost u frameworku, u kterého je obecně známo, že je poměrně náročný pro někoho, kdo s ním teprve začíná pracovat.

Popularita Foundationu ale v posledních letech opadá a od roku 2019 ho aktualizuje pouze komunita. Přední pozici v kategorii frameworků, které „nabízí všechno“, drží už poměrně dlouho Bootstrap. (Dokumentace Foundation, 2021)

# 4 Vzorový design WWW stránky pro zvolenou problematiku

Cílem praktické části této práce je především otestovat schopnosti jednotlivých knihoven vytvořit webové stránky na základě vzorového designu.

Samotný návrh je koncipován tak, aby odpovídal současným trendům a standardům z hlediska rozložení, typografických sekcí, grafických prvků (fotografie a ikonky), jednoduchosti a přehlednosti. Obsahuje verzi mobilní a desktopovou. Verze pro tablety nebyla vytvořena, protože přebírá vzhled ze zmíněných variant (je možné použít oboje). Žádná ze zvolených knihoven by tedy neměla mít sebemenší problém tento úkol zvládnout.

Logo bylo vytvořeno v aplikaci Adobe Illustrator a celkový design webu pak v Adobe XD. Všechny použité fotografie, ikonky a další prvky jsou zcela bezplatné, i pro komerční účely a pochází z Unspash[22] (fotky) a z Google Icons[23].

## 4.1 Popis layoutu a jednotlivých sekcí

Jako téma webu byl zvolen generický single-page pro fiktivní společnost Lorem Ipsum s.r.o. Layout je běžný a dnes často používaný. Nejvýš na stránce je navigační panel s logem a menu. Pod ním je sekce „O nás“, která obsahuje velký nadpis, kratší paragraf textu, informace o využitém frameworku, 2 tlačítka a větší obrázek. Sekce „Služby“ obsahuje 3 složené elementy (ikony, nadpis, krátký odstavec textu). Pod tím se nachází sekce „Portfolio“, ve které jsou 2 řádky po třech obrázcích. Pod každým obrázkem je stručný, jednořádkový popis. Dále je sekce „Reference“, která má v sobě 2 karty, a v každé z nich je fotografie a 2 různé druhy textu. Poslední sekce má název „Kontakty“ a její součásti je jednoduchý formulář. Na prvním řádku se nachází input pro jméno a pro e-mailovou adresu, na druhém pak pole pro zprávu. Pod formulářem je tlačítko na odeslání. Nejníž na stránce je footer, který je tmavý. Je zde opět logo (ale tentokrát v bílé variantě pro lepší čitelnost) a text s informacemi o použitém frameworku.

Náhled designu je v přílohách 2 (desktop) a 3 (mobilní zařízení).

[22] Unsplash: https://unsplash.com/
[23] Google Fonts Material icons: https://fonts.google.com/icons?selected=Material+Icons

# 5 Funkční prototypy webové stránky na základě navrženého vzoru s použitím vybraných frontendových knihoven

Cílem této kapitoly je představit informace a údaje o vytvořených prototypech a jejich srovnání na základě několika podstatných a běžně měřitelných faktorech.

## 5.1 Tvorba prototypů

Implementace všech frameworků do projektů proběhla hlavním, doporučeným způsobem, a také s ohledem na potřeby pro konkrétní problematiku. Při tvorbě každé verze byl použit verzovací systém git. Zdrojové kódy jsou tudíž k dispozici formou veřejných repozitářů na mém soukromém GitHub profilu.[24] Výsledné webové stránky jsou hostovány na fakultním serveru Kraken.[25] Konkrétní odkazy jsou uvedeny ke každé variantě, v podsekcích níže.

Kód byl psán v open source textovém editoru Visual Studio Code a částečně i ve vývojářských nástrojích prohlížečů Chromium a Mozilla Firefox Developer Edition. Všechny prototypy byly otestovány celkem na sedmi zařízeních / prostředích:

1. Linux desktop (3840x2160)
2. Linux notebook (1920x1080)
3. Windows 10 desktop (3840x2160)
4. Windows 10 notebook (1920x1080)
5. Android smartphone
6. Apple iPad
7. Apple iPhone

Na Linux a Windows zařízeních proběhly testy v prohlížečích Chromium (v91.0) a Mozilla Firefox (v89.0.2). Na Android smartphonu byly využity

[24] GitHub repozitáře: https://github.com/Golotvinov?tab=repositories
[25] Adresář na serveru Kraken: https://kraken.pedf.cuni.cz/~golotvia/BC_Prace/

Bromite (v90.0) a Mozilla Firefox (v89.1.1). Na Apple zařízeních byl použit výchozí prohlížeč Safari (v14).

Při tvorbě webů s vybranými frameworky nebyly použity žádné jiné knihovny nebo pomocné nástroje. Byly použity výchozí breakpointy a, kde to bylo možné a vyhovující, výchozí styly frameworku. Není nutno replikovat vzorový návrh 1:1, ale je potřeba se mu co nejvíce přiblížit a přenést ho do funkční podoby co nejvěrněji (struktura, rozložení a celkový designový jazyk se však musí zachovat).

Ke všem variantám byl přidán modal u tlačítka na odeslání formuláře. Aby mohl správně fungovat byl přidán malý kus JavaScriptu. Jelikož Bootstrap již tuto komponentu podporuje od základu, byl k této verzi připojen i komprimovaný JS soubor (bootstrap.min.js).

**Bootstrap verze**

- GitHub repozitář: https://github.com/Golotvinov/Bootstrap
- Odkaz na Kraken: https://kraken.pedf.cuni.cz/~golotvia/BC_Prace/Bootstrap/
- Způsob instalace: CDN

**Tailwind CSS verze**

- GitHub repozitář: https://github.com/Golotvinov/TailwindCSS
- Odkaz na Kraken: https://kraken.pedf.cuni.cz/~golotvia/BC_Prace/TailwindCSS/
- Způsob instalace: npm

**Pure verze**

- GitHub repozitář: https://github.com/Golotvinov/PureCSS
- Odkaz na Kraken: https://kraken.pedf.cuni.cz/~golotvia/BC_Prace/PureCSS/
- Způsob instalace: CDN

**Kontrolní verze (vanilla)**

- GitHub repozitář: https://github.com/Golotvinov/Vanilla

- Odkaz na Kraken:
  https://kraken.pedf.cuni.cz/~golotvia/BC_Prace/Vanilla/

## 5.2 Porovnání vytvořených prototypů

Všechny 3 frameworky si s úkolem poradily dobře, dle očekávání. I v takto poměrně jednoduchém a standardním případě se ale objevily v určitých kategoriích výrazné rozdíly.

### 5.2.1 Čas / doba vypracování

Významně se liší prototypy v tom, kolik času celkově zabralo je vyrobit. Měření bylo vždy zahájeno na začátku tvorby projektu, počínaje napsáním prvního řádku kódu v HTML souboru.[26] Do celkového času je započítaná veškerá práce kódování prototypu, včetně následných oprav chyb a dolaďování.

1. Bootstrap: 4h 51min
2. Tailwind CSS: 7h 52min
3. Pure: 8h 13min
4. Vanilla: 10h 01min

Nutno dodat, že se jedná o subjektivně zabarvenou kategorii, která se vztahuje k mým schopnostem a znalostem efektivně pracovat s jednotlivými technologiemi.

### 5.2.2 Rychlost načítání

Rychlosti načítání stránky byly změřeny v Network sekci vývojářské konzole prohlížeče Brave (Chromium engine). Cacheování bylo vždy vypnuté. Pro větší přesnost proběhlo měření každé verze webové stránky 5krát a výsledná doba je složena z průměru naměřených hodnot. Údaje v tabulce jsou uvedeny v milisekundách.

[26] U Tailwind CSS verze je v rámci férovosti započítán i čas instalace a přípravy projektu (odhadem +10 min).

*Tabulka 4 Srovnání rychlosti načítání prototypů v prohlížeči (hodnoty v ms)*

| | Bootstrap | Tailwind CSS | Pure | Kontrolní verze |
|---|---|---|---|---|
| 1. měření | 178 | 205 | 201 | 215 |
| 2. měření | 177 | 164 | 178 | 161 |
| 3. měření | 171 | 148 | 162 | 159 |
| 4. měření | 168 | 151 | 179 | 200 |
| 5. měření | 186 | 153 | 189 | 156 |
| **Průměrná doba načtení** | **176** | **164** | **182** | **178** |

Co je velmi zajímavé (a nečekané) je, že jsou všechny hodnoty velmi podobné. I přesto, že Pure a Bootstrap se načítají pomocí CDN, což představuje další požadavek na server (na jiný, než na kterém jsou hostovány).

### 5.2.3 Objem dat

Celková velikost stažená prohlížečem (-n-kB transfered over network), zahrnuje všechny soubory včetně obrázků. Čísla se značně liší u zdrojových adresářů Bootstrapu a Pure verze jelikož se samotné knihovny načítají z externího zdroje (CDN).

1. Pure: 417kB
2. Vanilla: 427kB
3. Tailwind CSS: 430kB
4. Bootstrap: 448kB

### 5.2.4 Dodatečný CSS kód

Pro přesnější napodobení navrženého vzoru bylo potřeba u všech prototypů s knihovnou dopsat nějaké vlastnosti v CSS.

U Bootstrapu bylo nutné přidat pod 30 řádků kódu. Verze s Pure obsahuje skoro 380 řádků CSS navíc. Vzhledem k účelu knihovny je to ale normální. Naprostou většinu stylů si kodér musí napsat sám. U Tailwindu stačilo 10 řádků kódu navíc. Verze bez knihoven (Vanilla) má pro srovnání skoro 600 řádků CSS.

# Závěr

V současnosti je k dispozici mnoho webových frameworků na tvorbu prezentační části webu. Velký výběr znamená, že existuje pro každého něco, ale také, že může být náročné se v tomto odvětví orientovat. O těch nejpoužívanějších je z velké části tato práce. Svoji popularitu a oblíbenost si ve frontend komunitě získaly splněním několika zásad. Jsou flexibilní a dávají vývojářům a designerům možnosti si je přizpůsobit podle svých potřeb, resp. podle potřeb projektu na kterém právě pracují. Dodržují doporučení a nařízení na tvorbu moderního webu (responzivní design, relativní jednotky, webová přístupnost, …). Jsou bezplatné, open source a na jejich vývoj se může podílet i komunita. V neposlední řadě je velmi podstatná kvalitní a detailně propracovaná dokumentace, která je napsána srozumitelně a obsahuje konkrétní příklady.

Kdy použít jaký framework (nebo za vůbec nějaký použít) se odvíjí hodně od toho co potřebujeme vyrobit a čeho chceme dosáhnout. Bootstrap je výborný pro případy kdy nemáme k dispozici žádný grafický návrh, ale chceme vytvořit web moderní a vizuálně příjemný. Nebo také na administraci či interní portály (např. pro zaměstnance) kde nehraje originalita vzhledu takovou roli. Ve výsledku je však často na první pohled znát, že byl použit Bootstrap. Na lidi, kteří se ve webovém vývoji nepohybují, může působit obyčejně a neoriginálně. Jeho výrazný a snadno rozpoznatelný design jistě už zná každý uživatel internetu. Práce s ním je rychlá a relativně jednoduchá, ale poznat všechny jeho funkce může chvíli trvat. Je možné ho začít efektivně používat i bez pokročilých znalostí ve webovém vývoji. Online je plno nástrojů a code snippets[27], kterými lze vývoj ještě více urychlit.

Tailwind CSS přináší úplně nový koncept. Je to framework, s jehož pomocí je možné vytvořit skoro jakýkoliv design. Na rozdíl od Bootstrapu či MaterialzeCSS nenabízí hotové komponenty a šablony. Uživatel si musí vše vyrobit sám a k tomu mu Tailwind dává nástroje formou utility tříd. Jedná se o časově náročnější framework, který je na začátku obtížnější na pochopení a používání. Výsledný HTML kód také nepůsobí líbivě, nejen kvůli velkému množství tříd, ale i kvůli faktu, že samotné třídy nejsou sémanticky pojmenovány.

[27] Hotové úryvky kódu, často nabízeny jako doplňky do textových editorů jako např. Visual Studio

Jakmile ale člověk pochopí jak správně pracovat s Tailwind CSS a začne využívat jeho možnosti, zjistí, že to nástroj pomocí kterého lze snadno a rychle vytvářet originální webové stránky s vlastním, komplexním i složitým designem, které jsou plně responzivní a vypadají dobře na všech prohlížečích.

Pure je ideální knihovna pro menší projekty, pro situace, kdy je priorita rychlost (ať už samotného vývoje nebo i načítání stránky), nebo kdy je potřeba co nejvíc šetřit objemem dat. Funguje spíše jako doplněk ke klasické tvorbě webdesignu než jako plnohodnotný webový framework. Kodér si styly tvoří sám a Pure pouze poskytuje základní řešení pro běžné či frustrující části, které neodmyslitelně patří k procesu tvorby webových stránek. Jeho layout systém je jednoduchý a efektivní, formuláře jsou pěkně zpracované a tlačítka, tabulky a navigační prvky mají aplikované základní styly pro následnou jednoduchou personalizaci. Velký potenciál vidím v jeho využití pro tvorbu uživatelských rozhraní aplikací v prohlížeči např. pro embedded zařízení. Projekt OpenWRT například používá ve své konfiguraci a možností nastavení routeru Bootstrap.

V dnešní době jsou webové frameworky pro všelijaké různé účely. Univerzální, jako ty, kterým se tato práce nejvíce věnuje, specializované (pouze na animace, na tlačítka, na typografii, na ikonky, …), s JavaScriptem i bez JavaScriptu a mnohé další. Jakou má tedy výhodu práce bez použití knihoven a kdy je lepší zvolit právě tuto variantu? Do doby než se objevil Tailwind se dělaly webové stránky, jejichž design nemohl být jednoduše napodoben v kódu za pomocí nějakého frameworku, bez využití podobných pomocných nástrojů.
S příchodem Tailwind CSS se však tyto situace staly vesměs minulostí. Otázka tedy není zda má vůbec smysl stavět web pomocí knihoven, ale s jakou knihovnou. Důležité jsou znalosti a zkušenosti programátora s daným nástrojem. (Michálek, 2021)

Každý framework má nějaká omezení. Práce v nich bude vždycky omezená možnostmi, které daný projekt nabízí. To je sice v pořádku, ale tyto limity je potřeba znát, než se rozhodneme svůj web postavit pomocí daného nástroje.

# Zdroje a použitá literatura


1. MICHÁLEK, Martin. Vzhůru do (responzivního) webdesignu. Verze 1.2. Praha: vlastním nákladem autora, 2018. ISBN 978-80-88253-00-6.

2. ŘEZÁČ, Jan. Web ostrý jako břitva: návrh fungujícího webu pro webdesignery a zadavatele projektů. Vydání druhé. [Brno]: House of Řezáč, 2016. ISBN 978-80-270-0644-1.

3. PILGRIM, Mark, [2015]. Ponořme se do HTML5. Praha: CZ.NIC, z.s.p.o. CZ.NIC. ISBN 978-80-905802-6-8.

4. GRANNEL Craig, SUMNER Victor, SYNODINOS Dionysios. The Essential Guide to HTML5 and CSS3 Web Design. Apress 2012. ISBN 978-1-4302-3786-0.

5. GOLDSTEIN, Alexis, Louis LAZARIS a Estelle WEYL. HTML5 a CSS3 pro webové designéry. Brno: Zoner Press, 2011. Encyklopedie webdesignera. ISBN 978-80-7413-166-0.

6. CEDERHOLM, Dan. Flexibilní webdesign: vytváříme přizpůsobitelné a přístupné stránky pomocí XHTML a CSS. Brno: Computer Press, 2006. ISBN 80-251-1018-4.

7. WOZNIEWICZ, Brandon. The Difference Between a Framework and a Library [online]. freeCodeCamp, 2019 [cit. 2021-6-20]. Dostupné z: https://www.freecodecamp.org/news/the-difference-between-a-framework-and-a-library-bd133054023f/

8. KOVÁŘ, Petr a Ondřej LETOCHA, 2021. Web Design Museum [online]. Praha [cit. 2021-6-21]. Dostupné z: https://www.webdesignmuseum.org/

9. The State of Developer Ecosystem 2020 [online], 2020. Praha: JetBrains [cit. 2021-6-21]. Dostupné z: https://www.jetbrains.com/lp/devecosystem-2020/

10. The Evolution of Web Design [online], c1995-2021. Harrisburg: WebFX [cit. 2021-6-19]. Dostupné z: https://www.webfx.com/blog/web-design/the-evolution-of-web-design/

11. HO TRAN, Tony, 2019. User-centered design: Definition, examples, and tips [online]. New York: InVisionApp [cit. 2021-6-20]. Dostupné z: https://www.invisionapp.com/inside-design/user-centered-design-definition-examples-and-tips/

12. DEVICES [online], 2021. Praha: Gemius [cit. 2021-6-23]. Dostupné z: http://ranking.gemius.com/cz/ranking/platforms/

13. Desktop vs Mobile vs Tablet Market Share Worldwide: May 2020 - May 2021 [online], 2021. Dublin: Statcounter [cit. 2021-6-23]. Dostupné z: https://gs.statcounter.com/platform-market-share/desktop-mobile-tablet/worldwide/#monthly-202005-202105-bar

14. 17 Best JavaScript CSS Framework Libraries: Curated by the Openbase team and community. [online], 2021. Severní Amerika: Openbase [cit. 2021-7-2]. Dostupné z: https://openbase.com/categories/js/best-javascript-css-framework-libraries

15. Mobile-first indexing best practices, 2016. In: Google Search Central [online]. Mountain View: Google, 2021 [cit. 2021-7-3]. Dostupné z: https://developers.google.com/search/mobile-sites/mobile-first-indexing

16. MICHÁLEK, Martin, 2015. Co je to „Mobile First“? Ale doopravdy. In: Vzhůru dolů [online]. Praha: Vzhůru dolů [cit. 2021-7-3]. Dostupné z: https://www.vzhurudolu.cz/prirucka/mobile-first

17. THOMSON, Greg, 2020. Web Accessibility Guidelines for Text. In: Accessibility for Ontarians with Disabilities Act [online]. Ontario: Accessibility for Ontarians with Disabilities Act [cit. 2021-7-4]. Dostupné z: https://aoda.ca/web-accessibility-guidelines-for-text/

18. Web Content Accessibility Guidelines (WCAG) 2.0: W3C Recommendation 11 December 2008, 2008. World Wide Web Consortium (W3C) [online]. Cambridge: World Wide Web Consortium (W3C) [cit. 2021-7-4]. Dostupné z: https://www.w3.org/TR/WCAG20/

19. GILLIS, Orlee, 2020. Web Typography: The Complete Guide for Designers. In: Elementor [online]. Israel: Elementor [cit. 2021-7-5]. Dostupné z: https://elementor.com/blog/guide-to-web-typography/

20. ELANSARY, Kareem, 2015. Font size, line height and line width the golden ratio way. In: Kareem Elansary – Medium [online]. Egypt: Medium [cit. 2021-7-5]. Dostupné z: https://medium.com/@zkareemz/golden-ratio-62b3b6d4282a

21. Blueprint CSS: A CSS framework that aims to cut down on your CSS development time. GitHub [online]. San Francisco: GitHub [cit. 2021-7-6]. Dostupné z: https://github.com/joshuaclayton/blueprint-css

22. Bootstrap 5 [online], 2021. San Francisco: Twitter Bootstrap [cit. 2021-7-6]. Dostupné z: https://getbootstrap.com/docs/5.0

23. Documentation - Tailwind CSS [online], 2021. Ontario: Tailwind CSS [cit. 2021-7-6]. Dostupné z: https://tailwindcss.com/docs

24. Pure [online], 2021. Sunnyvale: Pure [cit. 2021-7-6]. Dostupné z: https://purecss.io/

25. Vendor Prefix, 2021. In: MDN Web Docs [online]. San Francisco: Mozilla [cit. 2021-7-6]. Dostupné z: https://developer.mozilla.org/en-US/docs/Glossary/Vendor_Prefix

26. Best CSS Frameworks in 2021 [online], 2020. In: . Dev.To [cit. 2021-7-7]. Dostupné z: https://dev.to/theme_selection/best-css-frameworks-in-2020-1jjh

27. What are the best Front-End Frameworks Tools?, 2021. StackShare [online]. San Francisco: StackShare [cit. 2021-7-7]. Dostupné z: https://stackshare.io/front-end-frameworks

28. Wappalyzer, 2021. UI frameworks [online]. Melbourne: Wappalyzer [cit. 2021-7-7]. Dostupné z: https://www.wappalyzer.com/technologies/ui-frameworks

29. ČÁPKA, David, 2017. Lekce 1 - Úvod do CSS frameworku Bootstrap. In: Itnetwork.cz - Ajťácká sociální síť a materiálová základna pro C#, Java, PHP, HTML, CSS, JavaScript a další. [online]. Praha: ITnetwork [cit. 2021-7-7]. Dostupné z: https://www.itnetwork.cz/html-css/bootstrap/kurz/uvod-do-css-frameworku-bootstrap

30. Documentation: Bulma: Free, open source, and modern CSS framework based on Flexbox [online], 2021. London: Bulma [cit. 2021-7-8]. Dostupné z: https://bulma.io/documentation/

31. Documentation - Materialize [online], 2021. Pittsburgh: Materialize CSS [cit. 2021-7-8]. Dostupné z: https://materializecss.com/

32. The State of CSS 2020: CSS Frameworks: CSS Frameworks [online], 2020. [cit. 2021-7-9]. Dostupné z: https://2020.stateofcss.com/en-US/technologies/css-frameworks/

33. STRICKLAND, Jonathan, 2021. Is There a Web 1.0? In: HowStuffWorks [online]. HowStuffWorks [cit. 2021-7-12]. Dostupné z: https://computer.howstuffworks.com/web-10.htm

34. AGHAEI, Sareh, Mohammad Ali NEMATBAKHSH a Hadi Khosravi FARSANI, 2012. Evolution of the World Wide Web: From Web 1.0 to Web 4.0. In: University of Isfahan [online]. Isfahan: University of Isfahan [cit. 2021-7-12]. Dostupné z: http://www.cbpp.uaa.alaska.edu/afef/web%20evolution%201-4.htm

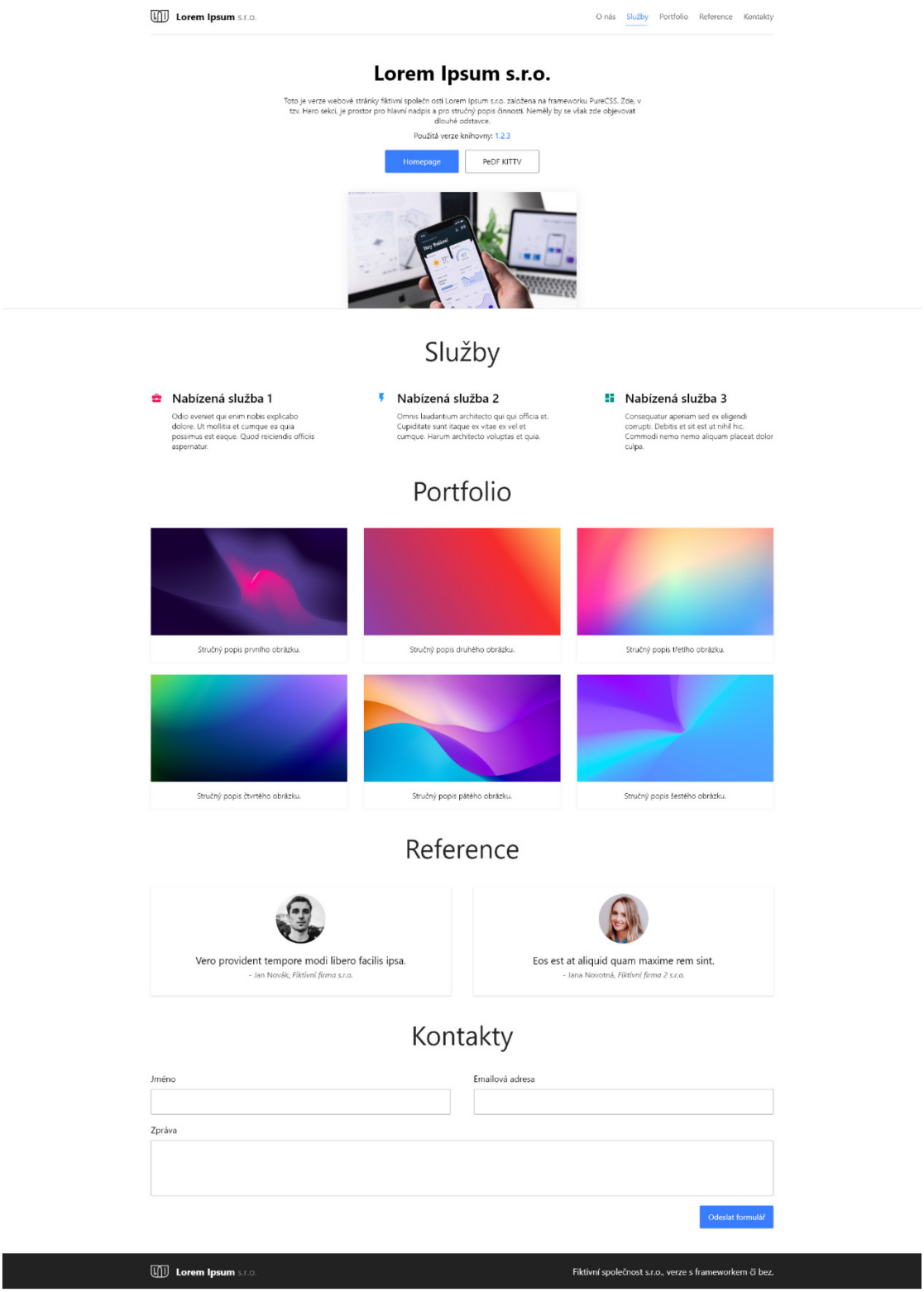
35. Téma: Frontendové frameworky
Rozhovor s Martinem MICHÁLKEM, Freelance page speed consultant, CSS expert. Praha 29.5.2021.

# Přílohy

## Příloha 1: Srovnání funkcí a možností vybraných knihoven

| | Bootstrap | Tailwind CSS | Pure |
|---|---|---|---|
| **Aktuální verze (při psaní práce)** | 5.0.2 | 2.2.4 | 2.0.6 |
| **Možnosti instalace / implementace v projektu** | | | |
| CDN | ✓ | ✓ | ✓ |
| Nord package manager (npm) | ✓ | ✓ | ✓ |
| Composer | ✓ | ✗ | ✓ |
| Stažení zdrojových souborů | ✓ | ✓ | ✓ |
| **Možnosti personalizace a vlastního nastavení** | | | |
| Import pouze potřebných části | ✓ | ✓ | ✓ |
| Odstranění nadbytečného / nepoužitého kódu | ✓ | ✓ | ✗ |
| Přidání vlastních stylů a proměnných (mimo vlastní CSS) | ✓ | ✓ | ✗ |
| **Layout možnosti** | | | |
| Mobile first | ✓ | ✓ | ✓ |
| Breakpoint úrovně | 6 | 6 | 5 |
| Vlastní layout engine / systém | ✓ | ✗ | ✓ |
| Možnost kombinovat s Flexbox a Grid | ✓ | ✓ | ✓ |
| **Typografie, barvy, styl** | | | |
| Základní úpravy textu | ✓ | ✓ | ✗ |
| Pokročilé typografické nástroje | ✓ | S oficiálním pluginem | ✗ |
| Barevná paleta | ✓ | ✓ | ✗ |
| Rozšířená barevná paleta | ✓ | ✓ | ✗ |
| Reset / sjednocení user agent stylů prohlížečů | ✓ | ✓ | ✓ |
| Silně nastylyzované / s vlastním designovým jazykem | Ano | Ne | Ne |
| Snadno přepsatelné výchozí styly | ✗ | ✓ | ✓ |
| **Předpřipavené komponenty** | | | |
| Základní (tlačítka, tabulky, navigace) | ✓ | ✗ | ✓ |
| Formuláře | ✓ | S oficiálním pluginem | ✓ |
| Pokročilé (kartičky, breadcrumbs, stránkování, ...) | ✓ | ✗ | ✗ |
| **Obsahuhje JavaScript** | Ano | Ne | Ne |
| **Dokumentace** | | | |
| Přehledná a srozumitelná dokumentace | ✓ | ✓ | ✓ |
| Oficiální video návody od vývojářů | ✗ | ✓ | ✗ |
| Hotové šablony | ✓ | ✗ | ✓ |
| **Obvyklá velikost v projektu** | 212KB (minified včetně JS) | ≤ 10KB (po použití PURGE) | 3.7KB |

# Příloha 2: Grafický návrh praktické části: Verze pro desktop



# Příloha 3: Grafický návrh praktické části: Verze pro mobilní zařízení