

VYSOKÁ ŠKOLA POLYTECHNICKÁ JIHLAVA

Aplikovaná informatika

Webová aplikace pro začínající programátory

Bakalářská práce

Autor práce: Petr Solař

Vedoucí práce: Mgr. Jana Valentová

Jihlava 2024

Vysoká škola polytechnická Jihlava

Tolstého 16, 586 01 Jihlava

ZADÁNÍ BAKALÁŘSKÉ PRÁCE

Autor práce: **Petr Solař**
Studijní program: Aplikovaná informatika
Obor: Aplikovaná informatika
Garant studijního programu: Ing. Lenka Kuklišová Pavelková, Ph.D.

Název práce: **Webová aplikace pro začínající programátory**

Vedoucí práce: Mgr. Jana Valentová

Cíl práce: Bakalářská práce se zabývá návrhem a implementací webové aplikace, která má za úkol seznámit uživatele se základy programování v jazycích C, C++, Java a tvorbou statických webových stránek. Kromě procházení jednotlivých lekcí online, mají uživatelé možnost stáhnout si vybranou tematiku ve formátu PDF do svého zařízení. Součástí aplikace je online kompilátor pro jazyky C, C++ a Java. Pro zájemce budou dostupné testy, které mohou obsahovat teoretické otázky, nebo fragment kódu, kde uživatel bude muset doplnit chybějící část.

Mgr. Jana Valentová
vedoucí bakalářské práce

Ing. Radek Kolman, Ph.D.
Katedra technických studií

Abstrakt

Bakalářská práce se zabývá návrhem a implementací webové aplikace, která má za úkol pomoci naprostým začátečníkům v oblasti programování. Návštěvníci mají možnost naučit se základy jazyků C, C++, Java, kromě programovacích jazyků mají uživatelé možnost seznámit se s tvorbou statických webových stránek. V úvodní části práce jsou popsány používané. V další části je popsán postup realizace webové aplikace. Výukový web byl vytvořený využitím znalostí z HTML, CSS, JavaScript, PHP a MySQL. Výsledkem práce je funkční web, který pomůže dostat se každému do světa programování.

Klíčová slova

Webová aplikace; php; výukový web; mysql; programování

Abstract

This bachelor's thesis deals with the design and implementation of a web application to help complete beginners in the field of programming. Visitors have the opportunity to learn the basics of C, C++, Java, in addition to programming languages, users have the opportunity to learn about creating static web pages. In the introductory part of the work, the technologies used are described. In the next part, the process of implementing a web application is described. The tutorial website has been created using knowledge of HTML, CSS, JavaScript, PHP and MySQL. The result of the work is a functional website that will help anyone to enter the world of programming.

Keywords

Web application; php; educational website; mysql; programming

Prohlašuji, že předložená bakalářská práce je původní a zpracoval/a jsem ji samostatně. Prohlašuji, že citace použitých pramenů je úplná, že jsem v práci neporušil/a autorská práva (ve smyslu zákona č. 121/2000 Sb., o právu autorském, o právech souvisejících s právem autorským a o změně některých zákonů, v platném znění, dále též „AZ“).

Byl/a jsem seznámen/a s tím, že na mou bakalářskou práci se plně vztahuje **AZ**, zejména § 60 (školní dílo).

Podle § 47b zákona o vysokých školách souhlasím se zveřejněním své práce podle směrnice prorektora pro studium č. 2/2020, a to bez ohledu na výsledek obhajoby.

Beru na vědomí, že VŠPJ má právo na uzavření licenční smlouvy o užití mé bakalářské práce a prohlašuji, že **s o u h l a s í m** s případným užitím mé bakalářské práce (prodej, zapůjčení apod.).

Jsem si vědom/a toho, že užít své bakalářské práce či poskytnout licenci k jejímu využití mohu jen se souhlasem VŠPJ, která má právo ode mě požadovat přiměřený příspěvek na úhradu nákladů, vynaložených vysokou školou na vytvoření díla (až do jejich skutečné výše), z výdělku dosaženého v souvislosti s užitím díla či poskytnutím licence.

V Jihlavě dne 30. ledna 2024

.....

Podpis studenta/ky

Poděkování

Chtěl bych poděkovat své vedoucí bakalářské práce paní Mgr. Janě Valentové, za cenné rady, trpělivost a ochotu při tvorbě této práce. A dále bych chtěl poděkovat své rodině za podporu a trpělivost.

Obsah

Seznam obrázků.....	7
Seznam zkratk.....	8
Úvod	9
1 Použité technologie	10
1.1 HTML.....	10
1.2 CSS	11
1.3 JavaScript	14
1.4 AJAX	15
1.5 Bootstrap	15
1.6 PHP.....	18
1.7 MySQL.....	18
1.8 XAMPP	19
2 Vývojové prostředí.....	20
2.1 StarUML	20
2.2 Visual Studio Code	20
2.3 phpMyAdmin	21
3 Praktická část	22
3.1 Use case diagram	22
3.2 Grafický návrh aplikace.....	23
3.3 Datový model.....	24
3.4 Možnosti aplikace	26
3.5 Online kompilátor	31
3.6 Zabezpečení aplikace	33
Závěr	34
Seznam použité literatury	35
Přílohy.....	36

Seznam obrázků

Obr. 1: HTML doplněná textem.....	10
Obr. 2: Ukázka stránky	11
Obr. 3: Externí CSS.....	12
Obr. 4: Externí soubor styly.css.....	12
Obr. 5: Inline CSS.....	13
Obr. 6: Interní CSS	13
Obr. 7: Vzhled webové stránky s použitím Inline CSS	14
Obr. 8: Ukázka kódu JavaScriptu	14
Obr. 9: Výsledek na stránce	15
Obr. 10: Přidání bootstrapu do projektu.....	16
Obr. 11: Bootstrap GRID systém	17
Obr. 12: Vzhled stránky s využitím bootstrapu	17
Obr. 13: PHP kód	18
Obr. 14: Výsledný vzhled.....	18
Obr. 15: XAMPP Control Panel.....	19
Obr. 16: Logo StarUML.....	20
Obr. 17: HTML dokument vytvořený ve VS Code.....	20
Obr. 18: Use Case Diagram vytvoření v programu StarUML.....	23
Obr. 19: Úvodní stránka	24
Obr. 20: Datový model aplikace	25
Obr. 21: Registrační formulář.....	26
Obr. 22: Menu pro přihlášeného uživatele	27
Obr. 23: Ukázka php pro vypsání sloupců do tabulky	29
Obr. 24: Ukázka výběru testů	30
Obr. 25: Probíhající test	30
Obr. 26: Vzhled kompilátoru	31
Obr. 27: Ukázka přepínání jazyků.....	32
Obr. 28: Ukázka jednoduchého programu.....	33

Seznam zkratek

HTML	Hypertext Markup Language (Hypertextový značkový jazyk)
CSS	Cascading Style Sheets (Kaskádové styly)
PHP	Hypertext Preprocessor (Hypertextový preprocesor)
MySQL	My Structured Query Language (Systém pro řízení databází)
IDE	Integrated Development Enviroment (Vývojové prostředí)
CDN	Content delivery network (Síť pro doručování obsahu)
AJAX	Asynchronous JavaScript and XML (Asynchronní JavaScript a XML)
DOM	Document Object Model (Objektový model dokumentu)

Úvod

V dnešní době, kdy máme spoustu informací a návodů týkajících se, jak začít s programováním. Ze začátku může nastat problém, že nevíme, kterou knížku, stránku nebo článek prostudovat jako první. Právě proto jsem se rozhodl pomoci s ulehčením výběru a vytvořit webovou aplikaci pro začínající programátory, která bude obsahovat návody pro široce využívané programovací jazyky jako je C, C++ a Java. Webová aplikace bude dále nabízet návody pro tvoření statických webových stránek pomocí HTML a CSS. Součástí aplikace bude online konzole, která uživatelům umožní vyzkoušet jednoduchý kód, se kterým se můžou při své cestě k úspěchu potkat bez nutnosti instalace externích aplikací.

Toto téma bakalářské práce mě napadlo, když jsem slyšel od svého bratrance stěžování o tom, jak je na základní škole vyučují programování a že materiály dostupné na internetu jsou z vysoké části pouze v anglickém jazyce. Přestože mi nebylo jasné, proč by zrovna studenti na základních školách měli znát základy programování, vnuknulo mi to nápad, že bych studentům v takové pozici mohl ulehčit trápení vytvořením webové aplikace, která by obsahovala jednoduchý úvod do programování v českém jazyce s názornými obrázky a texty.

Jako hlavní cíl práce považuji vytvořit aplikaci, která bude uživatelský přívětivá a pomůže studentům na základních školách, nebo komukoliv, kdo by měl zájem začít se vzdělávat v oblasti programování.

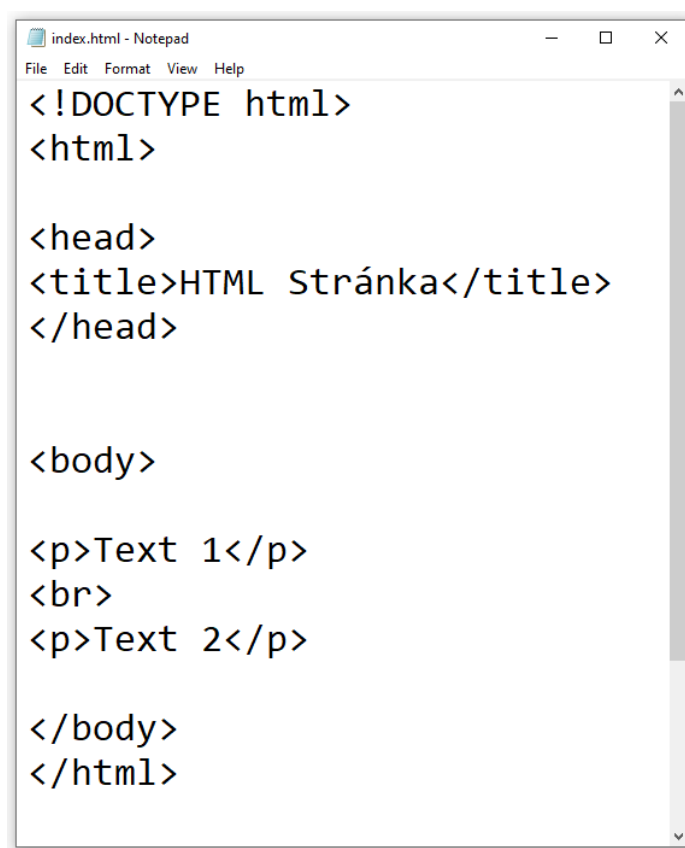
Při vybírání postupu při realizaci jsem se rozhodoval mezi dvěma způsoby, využití frameworku .net, který je široce využíván pro tvorbu a údržbu webových aplikací například pro programátory známý web stackoverflow. Druhá možnost byla využít PHP, se kterým jsem se při studiu na vysoké škole potkával víceméně denně. Přestože použitím frameworku .net by vývoj webu byl jednodušší a bezpečnější, rozhodl jsem se při svém projektu využít PHP, přeci jenom je práce zaměřená na základy programování a PHP patří mezi základní programovací jazyky používané při tvorbě webových stránek.

1 Použité technologie

Tato kapitola je zaměřená na popsání jednotlivých technologií, které jsem využíval při návrhu a implementaci webové aplikace. Kromě informací o použitých technologiích jsou kapitoly doprovázeny názornými ukázkami, jak by kdokoliv mohl vyzkoušet vytvořit jednoduchou vlastní webovou stránku s využitím pouze poznámkového bloku, který má každý uživatel operačního systému Windows k dispozici.

1.1 HTML

HyperText Markup Language, zkráceně HTML je jedním z nejpoužívanějších jazyků pro vývoj webových stránek, nejedná se však o programovací jazyk, ale jak nám název napovídá jedná se o značkovací jazyk. HTML nám umožňuje vytvářet čitelnou podobu webových stránek za pomoci tagů ať už párových nebo nepárových, rozdíl mezi párovými a nepárovými tagy vysvětlím na jednoduché ukázce.

A screenshot of a Notepad window titled 'index.html - Notepad'. The window contains the following HTML code:

```
<!DOCTYPE html>
<html>

<head>
<title>HTML Stránka</title>
</head>

<body>

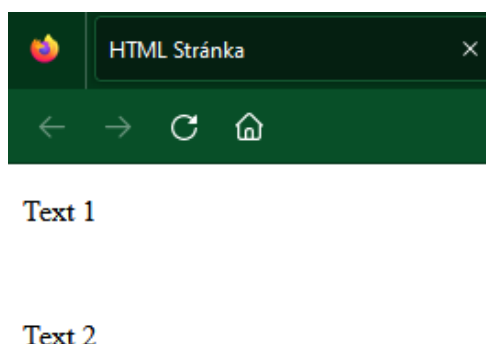
<p>Text 1</p>
<br>
<p>Text 2</p>

</body>
</html>
```

Obr. 1: HTML doplněná textem
(vlastní zpracování)

Na obrázku si můžeme všimnout, že pro vytvoření základní stránky za pomoci HTML jsem nepoužil nic jiného než poznámkový blok. Pokud se podíváme na strukturu můžeme si všimnout, že v prvním řádku dokumentu máme deklaraci, ta říká prohlížeči, jaký typ dokumentu má očekávat. V našem případě se jedná o `<!DOCTYPE html>`, který značí, že se jedná o HTML 5.

Dále máme párový tag `<html></html>`, který nám slouží jako schránka pro všechno, co bude dokument obsahovat. Párový tag znamená, že začíná otevřením tagu a zavřením tagu pomocí „/“. Uzavřením tagu značíme, že jakýkoliv obsah, který se bude vyskytovat mezi párovými tagy bude končit zavřením tagu. Pokud bychom tag neuzavřeli, může dojít ke špatné funkci stránky. Když budeme dále pokračovat v ukázce, vidíme párový tag `<head></head>`, který označuje hlavičku dokumentu, hlavička je schránka pro metadata, v naší ukázce se jedná o nadpis dokumentu `<title></title>` neboli text, který se objeví na záložce v prohlížeči. Následuje párový tag `<body></body>`, mezi kterým se nachází největší část obsahu webových stránek, pro naše tělo dokumentu obsahuje dva texty ohraničené párovým tagem `<p></p>` značící paragraf. Mezi textem 1 a textem 2 se vyskytuje nepárový tag `<br>`, který značí konec řádku neboli break line.



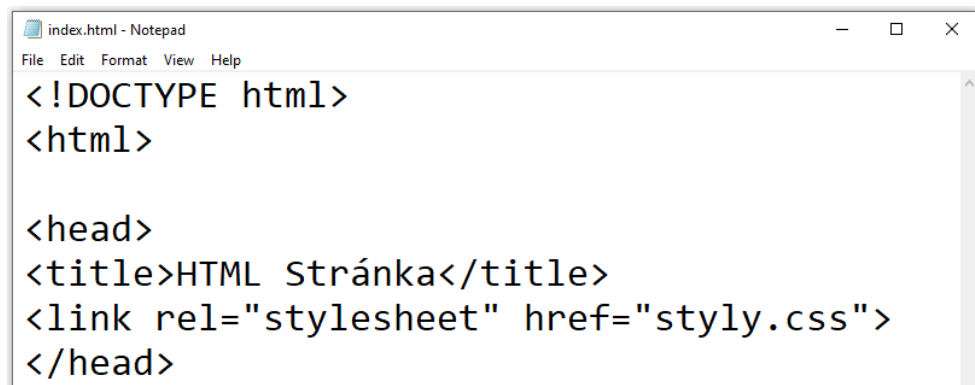
Obr. 2: Ukázka stránky
(vlastní zpracování)

1.2 CSS

Cascading Style Sheets, zkráceně CSS tvoří společně s HTML základ tvorby webových stránek. Pomocí CSS máme možnost nastavit, jak bude vypadat naše webová stránka, jakou barvu bude mít písmo na naší stránce a jak budou jednotlivé sekce umístěny. CSS můžeme na našich stránkách uplatnit třemi různými způsoby.

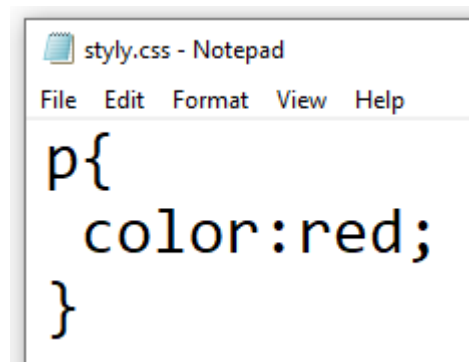
1.2.1 Externí CSS

Prvním a zároveň nejpoužívanějším způsobem, jakým můžeme ovlivnit vzhled naší stránky je pomocí externího souboru, v našem případě se jedná o soubor s názvem `style.css`. Po vytvoření souboru `style.css`, musíme do hlavičky dokumentu přidat odkaz na daný soubor.



Obr. 3: Externí CSS
(vlastní zpracování)

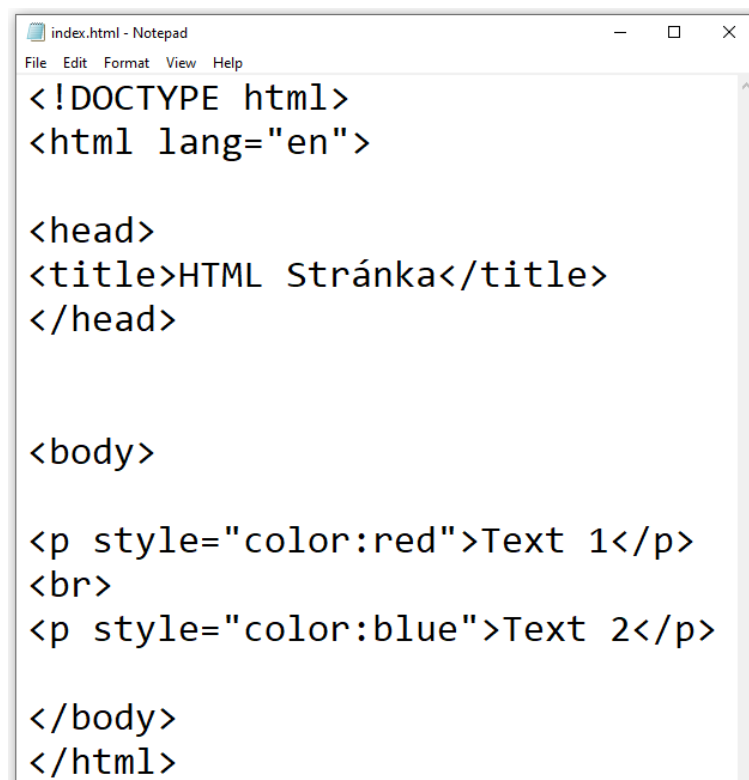
Nyní máme spojený soubor obsahující styly pro naši webovou stránku, můžeme tedy začít upravovat vzhled, pro ukázkou jsem si vybral změnu barvy textu u všech tagů typu <p>, v našem případě se jedná o Text 1 a Text 2, který jsme mohli vidět na obrázku výše.



Obr. 4: Externí soubor style.css
(vlastní zpracování)

1.2.2 Inline CSS

Druhým způsobem je definování stylu přímo u jednotlivých tagů, v tomto případě jsem pro Text 1 nastavil barvu textu červenou a pro Text 2 jsem nastavil modrou barvu.



```

index.html - Notepad
File Edit Format View Help
<!DOCTYPE html>
<html lang="en">

<head>
<title>HTML Stránka</title>
</head>

<body>

<p style="color:red">Text 1</p>
<br>
<p style="color:blue">Text 2</p>

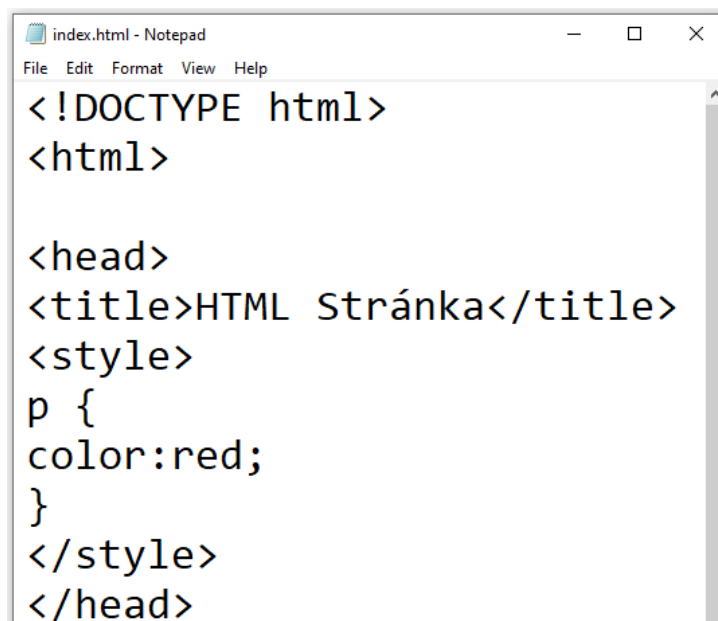
</body>
</html>

```

Obr. 5: Inline CSS
(vlastní zpracování)

1.2.3 Interní CSS

Třetím a posledním způsobem je využitím interního CSS, kde stylování stránky provádíme stejně jako u Inline CSS v souboru s obsahem pro naše webové stránky. Rozdíl mezi interním CSS a inline CSS je takový, že interní CSS se vkládá do hlavičky souboru a inline CSS se píše přímo k jednotlivým tagům.



```

index.html - Notepad
File Edit Format View Help
<!DOCTYPE html>
<html>

<head>
<title>HTML Stránka</title>
<style>
p {
color:red;
}
</style>
</head>

```

Obr. 6: Interní CSS

(vlastní zpracování)

Text 1

Text 2

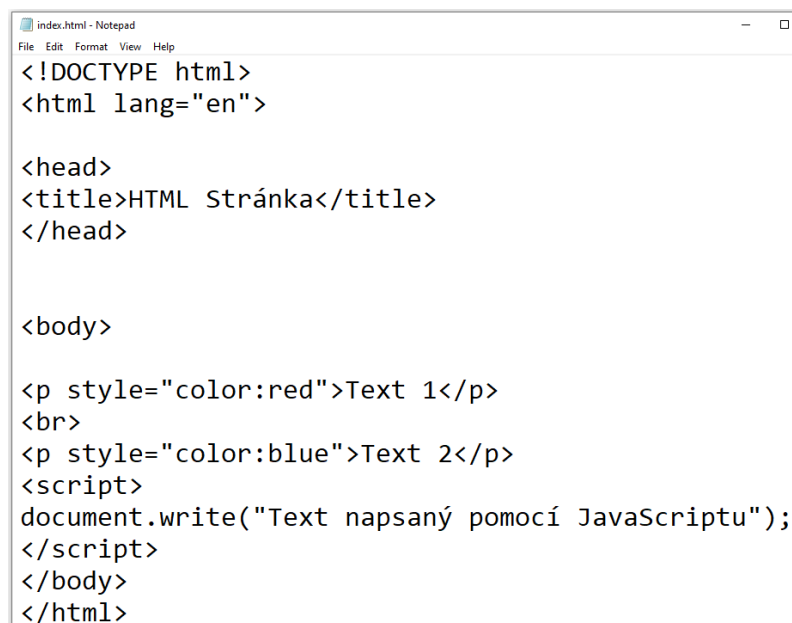
Obr. 7: Vzhled webové stránky s použitím Inline CSS

(vlastní zpracování)

1.3 JavaScript

JavaScript je skriptovací jazyk, který se využívá pro modernizaci dynamických webových stránek. Počátky sahají do devadesátých let dvacátého století, kdy se začal rozšiřovat z důvodů usnadnění ovládání a orientování na statických webových stránkách, které v té době byly velice běžné. (rascasone, 2022)

Programy neboli skripty se vkládají přímo do HTML kódu, jak k tomu můžeme vidět na dalším obrázku, pro jednoduchost náš skript pouze vkládá text, ale možností je spousta. Jedním z příkladu využití JavaScriptu je například kontrola webových formulářů.



```
index.html - Notepad
File Edit Format View Help
<!DOCTYPE html>
<html lang="en">

<head>
<title>HTML Stránka</title>
</head>

<body>

<p style="color:red">Text 1</p>
<br>
<p style="color:blue">Text 2</p>
<script>
document.write("Text napsaný pomocí JavaScriptu");
</script>
</body>
</html>
```

Obr. 8: Ukázka kódu JavaScriptu

(vlastní zpracování)

Text 1

Text 2

Text napsaný pomocí JavaScriptu

Obr. 9: Výsledek na stránce
(vlastní zpracování)

1.3.1 jQuery

Při vytváření webové aplikace jsem se setkal s příklady, které byly velice obtížné řešit prostým JavaScriptem a rozhodl jsem se tedy zapojit JavaScriptovou knihovnu jQuery, která má za účel zjednodušit implementaci JavaScriptu na webových stránkách. Velkým přínosem jQuery je zjednodušení některých programovacích částí jako například manipulace s CSS a reagování na události od uživatele jako například stisknutí tlačítka.

1.4 AJAX

Jelikož stránky mají za úkol poskytovat uživatelům možnost vyzkoušet si nabyté znalosti pomocí testů, které se skládají z jednotlivých otázek a uživatelé mají možnost přepínat mezi jednotlivými otázkami společně s automatickým odesláním testu po vypršení stanoveného limitu byl důležité zvolit vyhovující řešení. Jedno z možných řešení bylo využití AJAXu. AJAX pomáhá dynamicky měnit obsah stránek bez nutnosti znovu načítat danou stránku. Princip spočívá v asynchronním zpracování dat a následné manipulaci s DOM, stránka tak může reagovat na různé události bez přerušování běžících procesů.

1.5 Bootstrap

Přestože jsem se při tvorbě back endu pro webovou aplikaci odklonil od použití frameworků, tak pro vzhledovou část jsem se rozhodl použít framework bootstrap, který se využívá při tvorbě front endu. Bootstrap využívá systém dvanácti sloupců, funguje tedy tak, že stránka je rozložena do rovnoměrně širokých sloupců a my si podle potřeby můžeme nastavit, kolik sloupců má jednotlivá část našeho webu zaujmout.

Bootstrap je volně dostupný CSS framework, který vznikl za účelem ulehčení práce během vytváření grafické části webových aplikací. Bootstrap obsahuje předem definované styly upravující vzhled tabulek, tlačítek nebo například základního menu aplikace. Z toho může plynout problém, že všechny stránky vytvořené pomocí bootstrapu vypadají podobně, pokud tedy se tedy chceme odlišit od ostatních, je potřeba si styly ručně dopracovat za pomoci vlastního CSS. (Itnetwork, nedatováno)

Abychom mohli bootstrap použít v našem vytvořeném projektu, musíme ho nejdříve přidat. Na výběr máme ze dvou možností, bootstrap je distribuován jako balík souborů .css a .js, který můžeme stáhnout a následně přidat do našeho projektu anebo využít CDN. V naší ukázce jsem zvolil možnost využít CDN. Podobně jako u externího css přidáme do hlavičky našeho .html

souboru odkaz na styly určené pro bootstrap. Kromě odkazu na externí styly je také potřeba přidat script k načtení bootstrap JavaScript knihovny.

Pro ukázkou jsem vytvořil jednoduchý sloupcový systém a připojil ho k naší existující webové stránce. Můžeme si všimnout, že v prvním řádku jsou 3 stejně široké sloupce, které zabírají každý šířku 4 sloupců, ve druhém řádku můžeme vidět opět 3 sloupce, ale v tomto případě je první sloupec široký 4 sloupce, druhý sloupec 7 sloupců a na poslední sloupec nám podle bootstrapu zbývá pouze 1 sloupec, musíme tedy nastavit šířku jednoho sloupce, pokud bychom tak neudělali nebude bootstrap správně fungovat. Třetí a poslední řádek obsahuje jeden sloupec, který zabírá celou stránku a má tedy šířku 12 sloupců.

```
<!DOCTYPE html>
<html lang="en">

<head>
<title>HTML stránka</title>
<!--Bootstrap CSS-->
<link href="https://cdn.jsdelivr.net/npm/bootstrap@5.0.2/dist/css/bootstrap.min.css" rel="stylesheet"
      integrity="sha384-EVSTQN3/azprG1Anm3QDgpJLIm9Nao0Yz1ztcQTWfspd3yD65VohhpuuCOMLASjC"
      crossorigin="anonymous">
</head>

<body>

<p style="color:red">Text 1</p>
<br>
<p style="color:blue">Text 2</p>
<script>
document.write("Text napsaný pomocí JavaScriptu");
</script>
<!--Script k načtení knihovny-->
<script src="https://cdn.jsdelivr.net/npm/bootstrap@5.0.2/dist/js/bootstrap.bundle.min.js"
      integrity="sha384-Mrcw6ZMFYlzcLA8Nl+NtUVF0sA7MsXsP1UyJoMp4YLEuNSfAP+JcXn/tWtIaxVXM"
      crossorigin="anonymous"></script>

</body>
</html>
```

Obr. 10: Přidání bootstrapu do projektu
(vlastní zpracování)

```
<div class="container-fluid" style="border:solid 2px black">
  <div class="row">
    <div class="col" style="background-color:red">
      4 sloupce
    </div>
    <div class="col" style="background-color:green">
      4 sloupce
    </div>
    <div class="col" style="background-color:purple">
      4 sloupce
    </div>
  </div>
  <div class="row">
    <div class="col-4" style="background-color:yellow">
      4 sloupce
    </div>
    <div class="col-7" style="background-color:gray">
      7 sloupců
    </div>
    <div class="col-1" style="background-color:pink">
      1 sloupec
    </div>
  </div>
  <div class="row">
    <div class="col" style="background-color:maroon">
      12 sloupců
    </div>
  </div>
</div>
```

Obr. 11: Bootstrap GRID systém
(vlastní zpracování)

Text 1

Text 1

Text napsaný pomocí javascriptu

4 sloupce	4 sloupce	4 sloupce	
4 sloupce	7 sloupců		1 sloupec
12 sloupců			

Obr. 12: Vzhled stránky s využitím bootstrapu
(vlastní zpracování)

1.6 PHP

PHP, původně Personal Home page, nyní Hypertext Processor, je v dnešní době vysoce využívaný skriptovací jazyk, díky kterému můžeme v dnešní době tvořit moderní webové stránky. Díky PHP můžeme na stránkách měnit, ukládat či mazat data, například se může jednat o registraci uživatelů a následné přihlášení, pokud naše stránka neobsahuje registraci, můžeme php využít například k tvorbě ankety, nebo k monitorování návštěvnosti. Na rozdíl od JavaScriptu nebo HTML je potřeba PHP nainstalovat a nakonfigurovat, nejčastěji se tak provede s instalací Apache. (wikisofia, nedatováno)

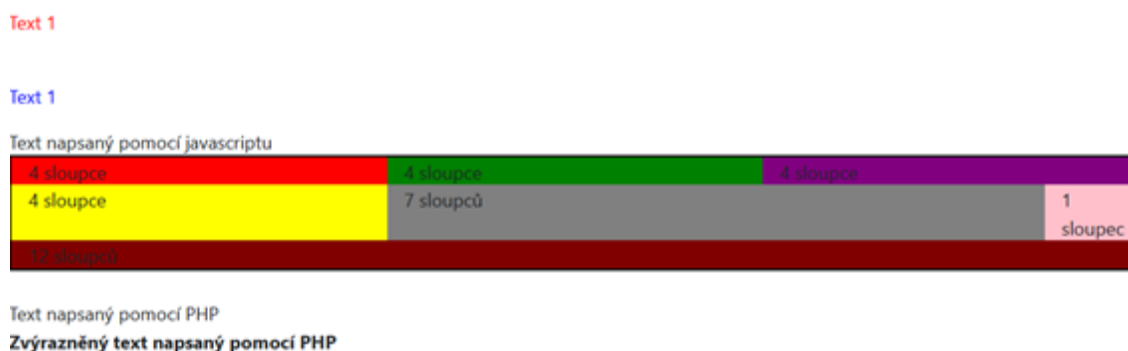
```
<?php

echo "<br>";
echo "Text napsaný pomocí PHP";
echo "<br>";
echo "<b>Zvýrazněný text napsaný pomocí PHP</b>";

?>
```

Obr. 13: PHP kód

(vlastní zpracování)



Obr. 14: Výsledný vzhled

(vlastní zpracování)

1.7 MySQL

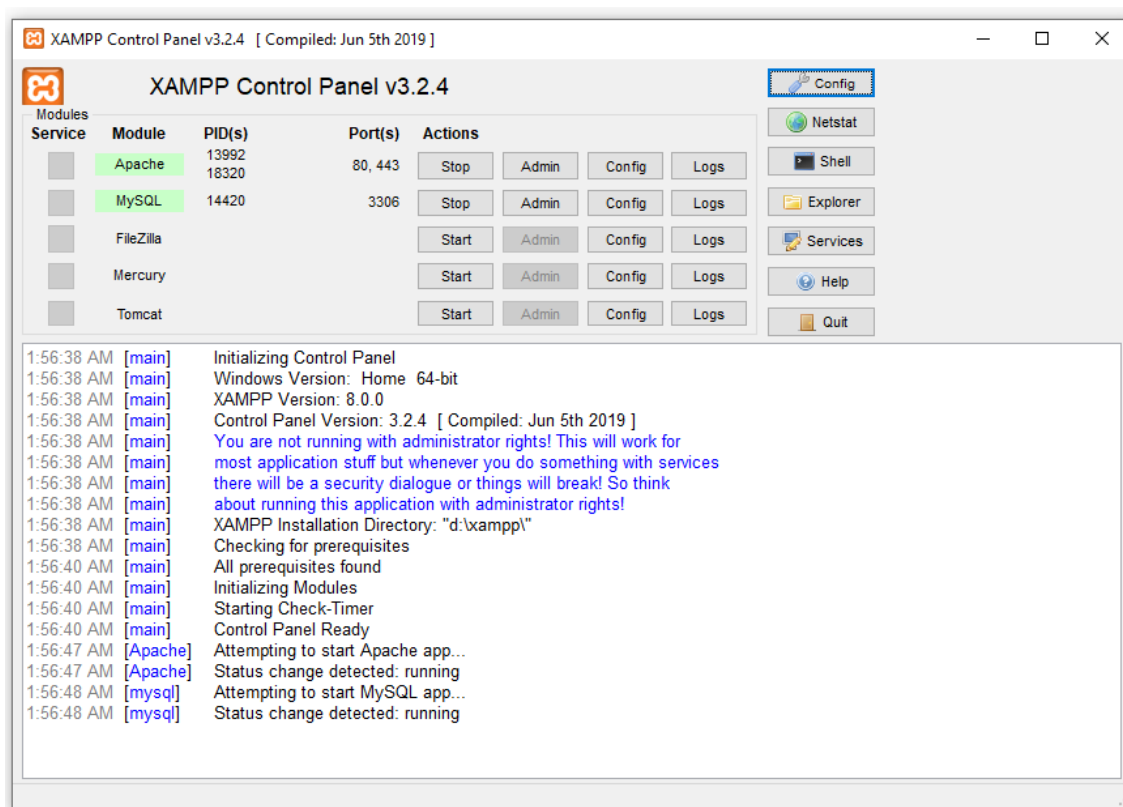
My Structured Query Language, neboli MySQL, je open-source systém pro správu a řízení databází. MySQL ukládá data v tabulkách, které jsou tvořeny jednotlivými sloupci a řádky, dle nastavení může mít tabulka až 1 073 741 824 řádků. Uživatelé mohou provádět dotazy neboli „queries“, mezi základní dotazy prováděné nad tabulkou patří select, update, delete a insert. (digitalocean.com, 2020)

1.7.1 MySQLi

V dnešní době je velice důležité klást důraz na bezpečnost a ochranu dat rozhodnul jsem se při práci použít MySQLi – My Structured Language Improved, je vylepšená varianta MySQL, která umožňuje efektivnější a bezpečnější komunikaci mezi programovacími jazyky, v mém případě PHP a MySQL databází. Mezi rozdíly patří například podpora objektově orientovaného programování nebo využití připravených dotazů, které pomáhají v prevenci proti sql injekcím.

1.8 XAMPP

Abych mohl testovat aplikaci na svém lokálním počítači bez využití webhostingu, rozhodl jsem se použít aplikaci XAMPP. XAMPP je volně dostupný software, která pomáhá jedincům vytvořit webový server na jakémkoliv lokálním počítači. Již název nám napovídá, co všechno XAMPP obsahuje, X značí, že je vhodný pro multiplatformní využití, A jako Apache server, M – MySQL, v novějších verzích bylo MySQL nahrazeno MariaDB, ale zkratka se zachovala, P pro PHP a poslední písmeno P jako Perl.



Obr. 15: XAMPP Control Panel

(vlastní zpracování)

2 Vývojové prostředí

Níže jsou uvedeny programy a nástroje, které jsem použil při realizaci webové aplikace. Hlavní část kódu byla napsána v programu Visual Studio Code, pro práci s databází byl využit phpMyAdmin a pro návrh a vytvoření use case diagramu společně s ER diagramem byl použit program StarUML.

2.1 StarUML

Přestože prostředí phpMyAdmin nabízí možnost vytváření E-R diagramů, rozhodl jsem se použít program StarUML, ve kterém jsem také vytvořil use case diagram, který znázorňuje, jak by měl systém fungovat.

StarUML je využíván především pro modelování a návrh systémů pomocí Unified Modeling Language, což je v softwarovém inženýrství grafický jazyk určený k vizualizaci, specifikaci a dokumentaci programových systémů. (staruml, nedatováno)



Obr. 16: Logo StarUML
(<https://staruml.io/>)

2.2 Visual Studio Code

Visual Studio Code (dále jen VS Code) je bezplatný editor zdrojového kódu. Přestože VS Code je vyvíjen společností Microsoft, je dostupný také na zařízeních využívajících Linux a macOS. Editor obsahuje podporu pro mnoho jazyků, jako např. Python, Ruby, Java, C++ a další. Uživatelům, kterým nestačí základní editor, mají možnost rozšířit svoje vývojové prostředí za pomoci volně dostupných balíčků, ty mohou pomáhat automaticky zarovnávat napsaný kód, nebo přidat podporu pro další programovací jazyk. (itnetwork, nedatováno)

```

index.html X
D: > bp > index.html > html > body
1  <!DOCTYPE html>
2  <html lang="en">
3  <head>
4    <meta charset="UTF-8">
5    <meta http-equiv="X-UA-Compatible" content="IE=edge">
6    <meta name="viewport" content="width=device-width, initial-scale=1.0">
7    <title>Document</title>
8  </head>
9  <body>
10 |
11 </body>
12 </html>

```

Obr. 17: HTML dokument vytvořený ve VS Code
(vlastní zpracování)

2.3 phpMyAdmin

Abych aplikaci mohl testovat na svém lokálním počítači, využil jsem nástroje phpMyAdmin, který je součástí programu XAMPP.

phpMyAdmin je volně dostupný software napsaný v PHP, který nám ulehčuje práci a administraci databáze MySQL a MariaDB pomocí webového rozhraní. Pokud používáme phpMyAdmina můžeme velice snadno provádět různé operace, které manipulují s databází nebo jednotlivými tabulkami v databázích, jako je například editace jednotlivých řádků tabulek, mazání celé tabulky nebo databáze. Mezi možnosti samozřejmě nechybí psaní vlastních SQL příkazů. V dnešní době je software jedním z nejpoužívanějších, co se týče správy a vytváření databází.

(phpmyadmin, nedatováno)

3 Praktická část

V následujících kapitolách se budu zabývat praktickou částí práce. Jako první si ukážeme, co můžeme od aplikace očekávat pomocí use case diagramu. Následovat bude popis a realizace grafického návrhu aplikace, návrh a ukázka řešení databáze a popis jednotlivých tabulek sloužících k chodu aplikace. Dále se dostaneme k registraci a autentizaci uživatelů, spouštění testů a realizaci online kompilátoru a v poslední části popíšu způsoby zabezpečení aplikace, kterými jsem se zabýval při vytváření webové aplikace.

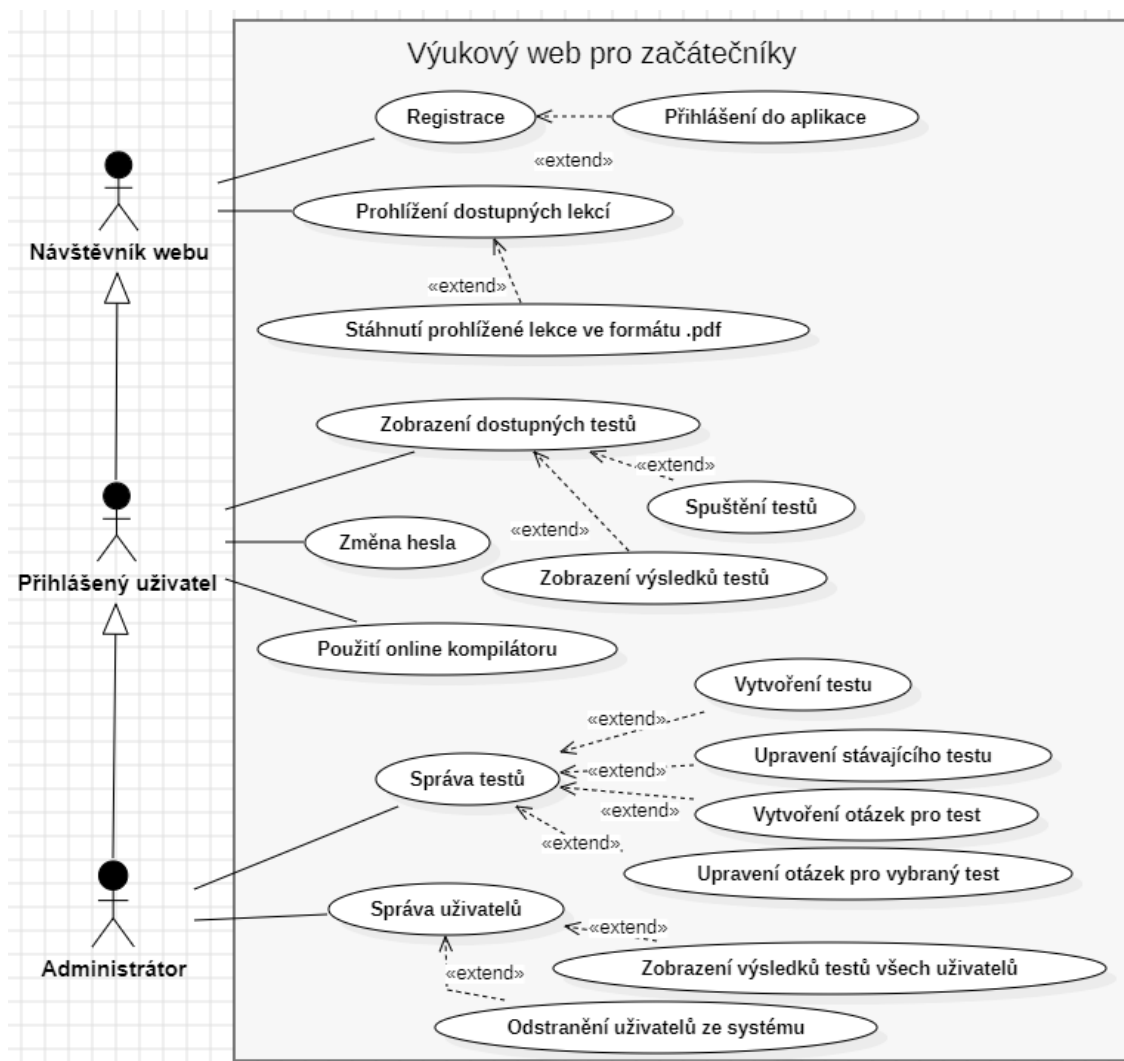
3.1 Use case diagram

Jakmile kdokoliv navštíví webové stránky, stává se z něj aktér v našem use case diagramu, pro naše stránky využijeme tři základní role, návštěvník webu, přihlášený uživatel a administrátor. Každá role má činnosti, které může vykonávat.

Základní rolí na webu je návštěvník webu, každý návštěvník webu má možnost registrovat se a po registraci se přihlásit. Po přihlášení se z návštěvníka webu stává přihlášený uživatel. Další možností pro návštěvníky je zobrazení dostupných výukových lekcí pro programovací jazyky C, C++ a Java a součástí jsou také lekce o základní tvorbě webu s využitím značkovacího jazyka HTML. Kromě prohlížení jednotlivých lekcí mohou také návštěvníci stáhnout každou lekci ve formátu .pdf.

Dalším aktérem v naší aplikaci je přihlášený uživatel, přihlášený uživatel má stejné možnosti jako návštěvník webu, a kromě toho má také další možnosti, které regulérní návštěvníci nemají. Přihlášení uživatelé mají možnost spouštět připravené testy k jednotlivým kapitolám a zobrazovat výsledky všech absolvovaných testů. Pokud by měli přihlášení uživatelé problémy s vlastním heslem, nebo se stalo že heslo ztratili či zapomněli, mají možnost změnit si heslo. Poslední možnost, kterou přihlášení uživatelé mohou využívat je dostupný online kompilátor, z bezpečnostních důvodů jsem používání online kompilátoru omezil pouze pro přihlášené uživatele.

Hlavním aktérem je v našem systému administrátor. Administrátor má všechny možnosti stejné jako návštěvníci webu a přihlášení uživatelé, ale také má možnost spravovat jednotlivé testy. To zahrnuje vytváření testů, úpravu stávajících testů, vytváření otázek pro daný test a úpravu stávajících otázek pro zvolený test. Kromě správy uživatelů má administrátor také možnost spravovat uživatele. Správa uživatelů zahrnuje odstranění uživatelů ze systému, nebo zobrazení všech výsledků uživatelů.



Obr. 18: Use Case Diagram vytvoření v programu StarUML
(vlastní zpracování)

3.2 Grafický návrh aplikace

Jak už bylo zmíněno v teoretické části práce, tak ke grafickému návrhu jsem využil technologii bootstrap a css. Při návrhu grafické aplikace jsem se hlavně zaměřil na to, aby aplikace byla přehledná a jednoduchá na navigování pro každého návštěvníka webu. V dnešní době samozřejmě nesmí chybět podpora pro mobilní zařízení, takže aplikace je responzivní a přizpůsobuje se velikosti zařízení, na kterém je zobrazena. Následující obrázek představuje, jak vypadá začátek úvodní stránky, pokud se návštěvník dostane na web. Jedná se o standardní layout, kdy po levé straně máme tlačítko domů a napravo menu, kde se uživatel může registrovat, přihlásit, nebo zvolit jednu z vybraných lekcí, které jsou dostupné. Pod lištou s menu máme uvítací zprávu, kde se návštěvník dozví, co může od webu očekávat a potom následují jednotlivé dostupné lekce. Položky menu se aktualizují podle toho, zda je uživatel přihlášen nebo nikoliv, pro kontrolu se využívá proměnných `$_SESSION` v php.



Lekce ▾
Přihlášení
Registrace

Vítej na stránkách pro nováčky v programování!

Chtěl by si se teleportovat do světa programování ale nevíš kde začít? V tom případě jsou tyto stránky právě pro tebe. Na stránkách najdeš materiály týkající se naprostých základů v programovacích jazycích **C**, **C++**, **Java**, nebo pokud by si se chtěl seznámit s tvorbou webů, jsou dostupné lekce, které obsahují základy značkovacího (né programovacího!) jazyka **HTML**. Nevíš kde začít? Zkus se podívat níže a přečti si něco o každém jazyku.

Programovací jazyk C

C(éčko), se začalo objevovat již v 70. letech a je považováno za předchůdce všech programovacích jazyků. Na základě jazyku C vzniklo velké množství dalších programovacích jazyků. C je velice rychlý a efektivní jazyk, který má mnoho využití.

- Operační systémy: C je základem operačních systémů jako UNIX, Linux a Windows.
- Vestavěné systémy: Mnoho vložených systémů, od mikrokontrolérů po zařízení IoT, je programováno v jazyce C.
- Vývoj her: Prvky populárních her, včetně Minecraftu a DooMu, byly vyvinuty v jazyce C.

Pojďme na to!

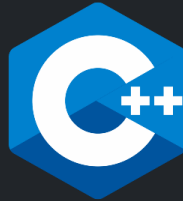


Programovací jazyk C++

Viděli jsme, C, teďka máme **C++**, jedná se o rozšíření, které má dodatečné funkce. C++ bylo představeno v 80. letech a oproti C klade důraz na objektově orientované programování narozdíl od strukturovaného.

- Stejně jako C, se C++ využívá pro vývoj moderních her.
- Systémový/softwareový software: Mnoho desktopových aplikací a softwarových systémů je vyvíjeno v jazyce C++.
- Aplikace pro grafiku a CAD: AutoCAD a systémy Adobe používají C++ pro svá grafická rozhraní.

Pojďme na to!

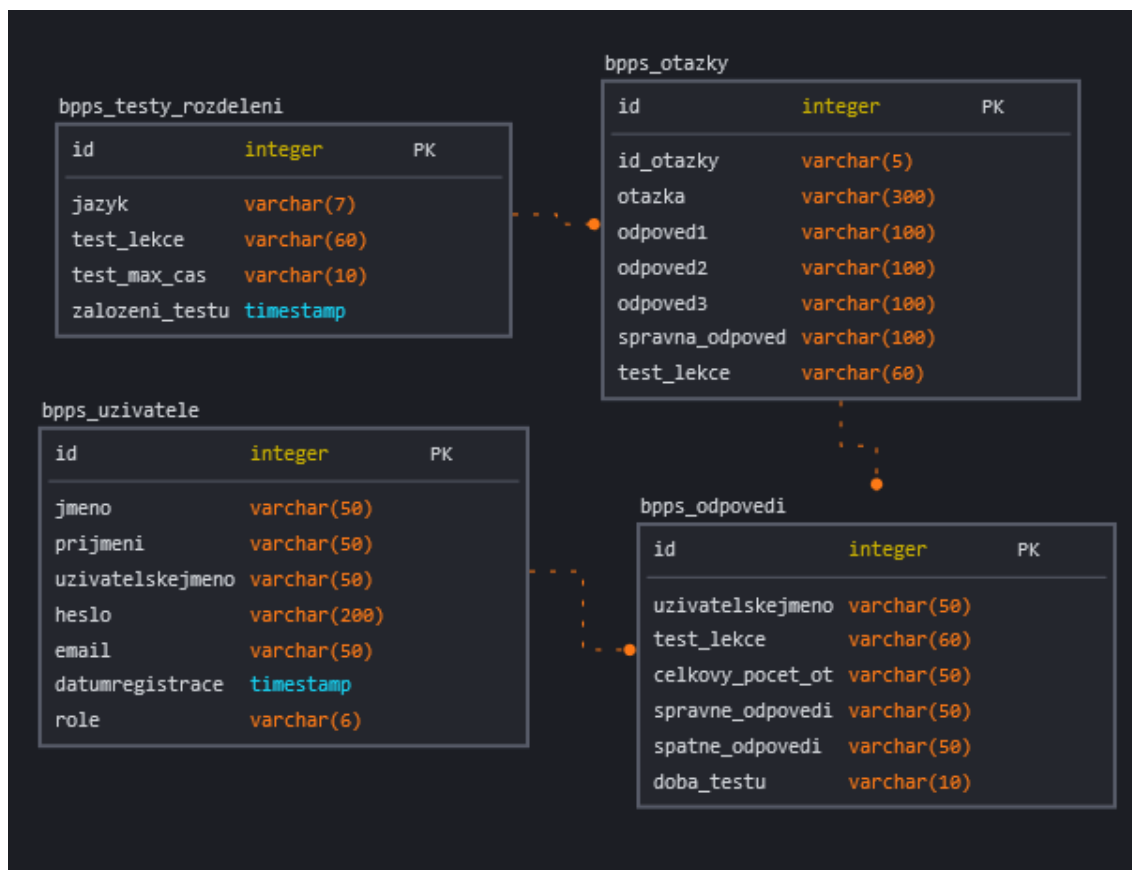


Obr. 19: Úvodní stránka
(vlastní zpracování)

Pro pokus o zvýšení pozornosti, jsem ke stylování tlačítek nepoužíval předdefinované styly, které jsou dostupné při používání bootstrapu, ale navrhnul jsem vlastní tlačítka, které mají jednoduchou animaci, když na ně uživatel najede myší.

3.3 Datový model

Pro uchovávání informací o uživateli, testech, jednotlivých otázkách a uchování výsledků jednotlivých testů aplikace využívá celkově čtyř tabulek.



Obr. 20: Datový model aplikace

(vlastní zpracování)

Ke správě uživatelů je připravená tabulka `uzivatele`, která má jako primární klíč `id`, který se automaticky inkrementuje s každým novým záznamem. Do dalších sloupců v tabulce se ukládá jméno, příjmení, uživatelské jméno, heslo a email, na který se uživatel zaregistroval. Všechny tyto pole se vyplní podle vstupu uživatele při registraci. Dalšími poli je datum registrace, která se vyplní automaticky v čas, kdy uživatel provede registraci a posledním sloupcem v tabulce je role, která přiřazuje defaultní roli – user.

Další důležitou tabulkou je tabulka pro uchování základních informací o jednotlivých testech `testy_rozdeleni`. V této tabulce jsou sloupce pro identifikaci jednotlivých testů, jako je jazyk, kterého se test týká, dále pak unikátní název testu, maximální čas stanovený na test a kdy byl test založen. Všechny informace kromě doby založení testu zadává administrátor při vytváření jednotlivých testů.

Pro jednotlivé otázky k testům je připravená tabulka `otazky`. Otázky jsou typu výběr ze tří odpovědí, v tabulce jsou tedy tři pole pro odpovědi, pole pro číslo otázky a správnou odpověď. Pro přiřazení otázek k testům se využívá pole `test_lekce`.

Aby si uživatel mohl zobrazit úspěšnost testů, které v minulosti vyzkoušel, je připravená tabulka `odpovedi`. Tabulka obsahuje sloupce jako název testu, uživatelské jméno a jakou úspěšnost měl uživatel při dokončení testu, posledním sloupcem je kdy se uživatel pokoušel test splnit.

3.4 Možnosti aplikace

3.4.1 Registrace

Uživatelé, kteří by chtěli vyzkoušet veškeré funkce webu jako je vyzkoušení online kompilátoru a procvičování testů z dostupných kapitol se nejdříve musí zaregistrovat. Při vytváření registrace jsem vybíral mezi dvěma možnostmi. Udělat separátní stránky pro přihlašování a registraci, nebo využít dnes široce využívaných vyskakovacích oken, které jsou součástí frameworku bootstrap. Jelikož jsem se při studiu přiklonil skoro vždy k vytváření separátních stránek, při vytváření bakalářské práce jsem si chtěl vyzkoušet a naučit se co nejvíce věcí a rozhodl jsem se tedy využít vyskakovacích oken.

Obr. 21: Registrační formulář
(vlastní zpracování)

Aby vyskakovací okna zapadaly vizuálně ke stránkám, musel jsem je dopravit s využitím css a atributu !important, který přepíše všechna základní nastavení. Uživatelé se můžou setkat celkem se třemi vyskakovacími okny, mezi kterými můžou cestovat využitím tlačítek v pravém a levém rohu, pro přehlednost a zdůraznění možnosti pro obnovení hesla je tlačítko zvýrazněno červeně.

Pokud by uživatel potřeboval zobrazit heslo, může využít checkbox s ikonou oka, která přepíná typ polí mezi password a text.

Jelikož se jedná o formulář a uživatel může zadávat cokoliv, probíhá před odesláním dat do databáze několik kontrol. Odesílání a kontrola dat probíhá pomocí jquery a ajaxu. Pokud jsou vyplněna všechna pole a klikne se na tlačítko „registrovat“ data z jednotlivých polí se zapíší do proměnných a začínají další kontroly, pokud se vyskytne problém v některých z vyplněných polí je o tom uživatel upozorněn pomocí chybových hlášek, aby si uživatel mohl přečíst, jak registrace probíhala musel jsem zastavit základní akci, která se provede po kliknutí na tlačítko ve

vyskakovacím okně, a to je automatické zavření okna. Pro zamezení automatického zavření okna jsem opět využil jquery. Ověřování dat je rozděleno na dvě části, v první části se kontrolují věci mimo databázi jako délka hesla, nebo zda jsou obě zadaná hesla stejná. V další části už je pokus o zapsání do databáze, kde se kontroluje například unikátnost uživatelského jména nebo emailu, pokud se nenajde žádná chyba jsou data odeslána do databáze pomocí připravených dotazů, které lze využít při používání MySQLi.

3.4.2 Autentizace uživatelů

Jakmile uživatel úspěšně dokončí registraci má možnost se přihlásit. Pokud uživatel právě dokončil registraci, může se do okna s přihlášením dostat kliknutím na tlačítko přihlášení v pravém rohu, jinak může využít tlačítka pro přihlášení v hlavním menu stránky. Pro přihlášení potřebuje uživatel vlastní uživatelské jméno a heslo, které zadal při registraci. Stejně jako u registrace se ověření skládá ze dvou částí, v první části se kontroluje, zda jsou vyplněná všechna pole, pokud jsou vyplněná obě pole probíhá kontrola v databázi. Kontroluje se, zda uživatel existuje a zda sedí zadaného hesla, jelikož se hesla ukládá pomocí php funkce password_hash, musí se shodování hesel kontrolovat pomocí funkce password_verify, která kontroluje, zda heslo odpovídá hashované verzi. Pokud přihlášení proběhne úspěšně, zakládají se proměnné \$_SESSION s uživatelským jménem a rolí uživatele a uživatel je pomocí javascriptu přesměrován na úvodní stránku pro uživatele. Úvodní stránka obsahuje uvítací zprávu a tabulku, která ukazuje výsledky posledních tří testů, které uživatel absolvoval. Kromě zprávy a tabulky s výsledky zde uživatel vidí email, na který je účet registrován a má zde možnost změnit vlastní heslo. Jakmile je uživatel přihlášen, změní se položky hlavního menu.



Obr. 22: Menu pro přihlášeného uživatele
(vlastní zpracování)

Místo přihlášení a registrace se v menu po přihlášení objeví tlačítka pro odhlášení, nástěnku neboli hlavní stránku pro přihlášené uživatele a v rozbalovacím menu se navíc kromě všech dostupných lekcí zobrazí sekce pro online testy. Kromě změny v hlavním menu se uživatel setká s postranním menu, kde má na výběr prohlédnout si dostupné testy, nebo zkontrolovat, jak dopadl ve všech testech, které kdy absolvoval.

Možnost procházení testů a využívání konzole je určeno pouze pro přihlášené uživatele, pokud by se nepřihlášený uživatel pokusil dostat na stránku s testy, nebo konzolí, bude automaticky přesměrován na hlavní stránku aplikace.

3.4.3 Obnovení hesla

Pokud by nastala situace, že uživatel zapomene heslo od registrovaného účtu, může v menu po kliknutí na přihlášení nebo registraci má možnost kliknout na tlačítko zapomenuté heslo, které přepne okno na formulář týkajícího se obnovy hesla. Pro obnovení hesla musí uživatel zadat emailovou adresu, na který je účet registrován. Stejně jako při registraci a autentizaci se využívá jquery a ajaxu pro zpracování požadavku a kontroly zadaných údajů. Pokud uživatel zadá platnou emailovou adresu obdrží na email nové vygenerované heslo, které se vytvoří pomocí funkce

s náhodným výstupem. Email se posílá pomocí funkce mail(). Jak už bylo uvedeno výše, uživatel má možnost změnit si heslo po přihlášení.

3.4.4 Stáhnutí .pdf

Možnost stáhnutí jednotlivých lekcí je dostupná pro každého návštěvníka webu. Pro stáhnutí stačí návštěvníkovi kliknout na tlačítko stáhnout na konci každé lekce, po kliknutí se lekce automaticky stáhne jako soubor s příponou .pdf a zachová původní vzhled. Pro realizaci jsem využil javascriptové knihovny html2pdf, která zajistí uložení vybraného elementu a veškerého obsahu do souboru. Tlačítka mají událost onclick, která spustí daný script.

3.4.5 Testy a otázky

V této kapitole si ukážeme vše týkající se dostupných testů a otázek. Možnost přidávat, upravovat a mazat testy a otázky má pouze admin. Stejně jako ostatní uživatelé se admin musí před prováděním jakýchkoliv operací přihlásit. Přihlašování probíhá stejně jako u běžných uživatelů, kde jedinou změnou je místo pro přesměrování, běžní uživatelé jsou přesměrováni na stránku pro uživatele a admin je přesměrován na stránku pro admina. Zda se jedná o admina nebo běžného uživatele určuje session proměnná role, která se zakládá při úspěšném přihlášení. Admin má stejné možnosti jako uživatel, co se týče prohlížení a startování testů a používání konzole, ale kromě těchto věcí má přístup ke správě testů a uživatelů. Správa testů spočívá založení jednotlivých testů, nechybí samozřejmě možnosti pro úpravu a smazání vybraného testu. Další možností je vytváření jednotlivých otázek pro vybraný test, při vytváření otázek může admin upravovat nebo mazat stávající otázky. Kromě vytváření testů a otázek může admin zobrazovat výsledky všech uživatelů, kteří se v nedávné době pokusili o vyplnění testu. Další možností je správa uživatelů, která spočívá v zobrazení všech registrovaných uživatelů a možností smazat vybraného uživatele.

Pokud admin chce vytvořit test, má k tomu připravený formulář, kde vybírá, jakého jazyka se test týká, na výběr má ze čtyř možností zobrazených pomocí elementu <select>, čím se test zabývá a jaká je povolená maximální délka pro test. Po vytvoření testu se data uloží do databáze a stránka se obnoví a vytvořený test je vidět v tabulce spolu s ostatními testy, kromě vyplněných informací o testu jsou v tabulce také pole pro úpravu nebo smazání testu.

Pokud admin klikne na úpravu testu, informace o vybraném testu se načtou do nového formuláře, ze kterého může admin test opravit. Pokud by kliknul na volbu pro smazání testů, zobrazí se nejdříve potvrzovací okno. Pokud admin potvrdí smazání, zvolený test společně se všemi otázkami, které jsou součástí testu odstraní.

Jakmile je vytvořený test, tak se dají vytvářet otázky. Pro vytváření otázek se admin musí překliknout v menu na stránku pro vytváření otázek, kde uvidí seznam všech testů a tlačítko pro výběr specifického testu. Po vybrání testu se admin dostane k formuláři, kde zadává znění otázky, tři možné odpovědi pro otázku a správnou odpověď. Stejně jako u testů výsledek uvidí okamžitě, protože stránka se automaticky obnoví po přidání otázky. Pokud dojde k vymazání některé z otázek z testu, provádí se při přidání každé otázky kontrola, zda má správné číslo otázky. Toto řešení obsahuje menší nevýhodu, pokud by se admin rozhodnul odstranit otázku, která není poslední v testu, musel by pro aktualizaci čísla otázky založit novou otázku a tu následně vymazat. Úprava jednotlivých otázek funguje stejně jako úprava testů, kdy se

informace o vybrané otázce objeví v novém formuláři, ve kterém následně může admin provádět změny.

Zobrazení výsledků testů je různé pro admina a běžného uživatele, admin má možnost zobrazit si výsledky všech uživatelů, běžný uživatel má možnost zobrazit si pouze vlastní výsledky. Výsledky testů se automaticky načítají pomocí php z tabulky v databázi. Pro lepší přehlednost jsou data zobrazená v tabulce, která má vzhled upravený pomocí bootstrapu.

```
$offset = 0;

if (isset($_GET['offset'])) {
    $offset = (int) $_GET['offset'];
}

$totalL = "SELECT * from bpps_odpovedi";
$runTotalL = mysqli_query($conn, $totalL);
$pocetTestu = mysqli_num_rows($runTotalL);

$showTestsQ = "SELECT * from bpps_odpovedi LIMIT 10 OFFSET $offset";
$runShowTestsQ = mysqli_query($conn, $showTestsQ);
while ($row = mysqli_fetch_assoc($runShowTestsQ)) {
    ?>
    <tr>
        <td>
            <?php echo $row["uzivatelskejmeno"]; ?>
        </td>
        <td>
            <?php echo $row["test_lekce"]; ?>
        </td>
        <td>
            <?php echo $row["celkovy_pocet_ot"]; ?>
        </td>
        <td>
            <?php echo $row["spravne_odpovedi"]; ?>
        </td>
        <td>
            <?php echo $row["spatne_odpovedi"]; ?>
        </td>
    </tr>
    <?php
}

?>
```

Obr. 23: Ukázka php pro vypsání sloupců do tabulky
(vlastní zpracování)

Mimo testy může admin zobrazit všechny uživatele, kteří jsou v současné době registrovaní, na rozdíl od testů zde není možnost pro úpravu vybraných uživatelů, pouze možnost pro smazání, pokud by se admin rozhodl odstranit nějakého uživatele, bude stejně jako u mazání testů nejdříve muset potvrdit akci. Pro potvrzení akcí ze strany admina jsem využil jednoduché javascript funkce confirm(), pokud se jedná o potvrzování ze strany uživatelé, používal jsem podobně jako u přihlašování a registrace modal okna.

Ve chvíli, kdy jsou vytvořené testy a otázky, mají možnost uživatelé spustit test, o který mají zájem. Pokud neexistují žádné testy, které by mohli uživatelé spouštět, namísto obvykle tabulky se objeví zpráva, která informuje zájemce o stavu testů.

Programovací jazyk	Obsah testu	Povolený čas na test	Tlačítko pro spuštění
C	Datové typy	5 minut	Datové typy
HTML	Tabulky	5 minut	Tabulky
C++	Cykly	10 minut	Cykly

Obr. 24: Ukázka výběru testů
(vlastní zpracování)

Jak vypadá tabulka s dostupnými testy je vidět na obrázku výše, uživatel snadno vidí, jakého jazyka se test týká, co je obsahem testu a povolený čas na test, pokud uživatel najede myší na tlačítko pro spuštění změní se pozadí tlačítka na zelené. Pokud uživatel klikne na tlačítko pro spuštění testu, objeví se ještě potvrzovací okno, jestli si skutečně přeje začít test. Na výběr má ze dvou možností, spustit test anebo zrušit, pokud klikne na tlačítko zrušit, zavře se potvrzovací formulář a uživatel může nadále prohlížet dostupné testy. Pokud uživatel potvrdí spuštění testu, stránka se přepne na běžící test. Kromě přepnutí na novou stránku se pomocí ajaxu nastaví session proměnná, která uchovává název testu a dále se používá pro načítání otázek, maximální povolené doby pro zvládnutí testu a spuštění časomíry.

Obr. 25: Probíhající test
(vlastní zpracování)

Kromě samotného názvu testu se při spuštění testu načte celkový počet otázek, jednotlivé otázky, mezi kterými jde proklikávat pomocí tlačítek. Pro načtení například celkového počtu otázek se využívá funkce `celkovyPocetOt()`, která aktualizuje obsah divu, v mém případě jednociferné číslo, s patřičným id, pro vypsání se používá div označený `id="celkovyPocetQ"`. Celkový počet otázek v testu je získán metodou GET ze souboru `nacteniPocetOt.php`, který za pomoci session proměnné, která uchovává název testu najde v patřičné tabulce kolik obsahuje test otázek. Na podobném principu funguje také načítání otázek. Pro odpočet času na test se využívá javascriptové funkce `setInterval`, která každých 1000 milisekund volá patřičnou funkci.

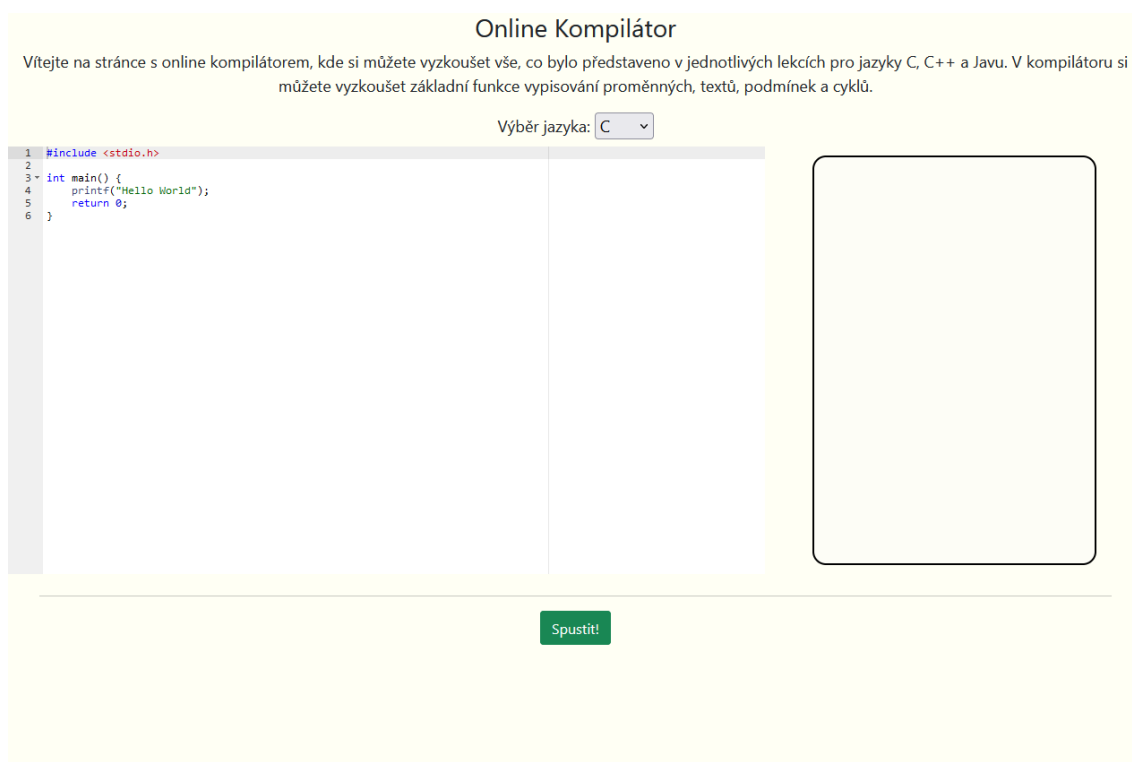
Tlačítka, která zacházejí s testem mají vlastní opatření, tlačítko, které slouží pro volání předešlé otázky neudělá nic, pokud je uživatel na první otázce, pokud uživatel klikne na tlačítko pro další otázku, když je na poslední otázce v testu, zobrazí se informační okno, že je na poslední otázce a zda si přeje ukončit test. Kromě ukončení testu klikáním v poslední otázce mají uživatelé také

možnost test ukončit okamžitě. Kromě manuálního ukončení se test odesílá k vyhodnocení v té době, kdy vyprší časový limit pro vybraný test.

Jakmile uživatel ukončí test, je přesměrován na stránku s výsledky, kde může vidět celkový počet otázek, na kolik otázek odpověděl správně a špatně a celkovou úspěšnost testu uvedenou v procentech. Součástí výsledků je tlačítko pro návrat na stránku s dostupnými testy.

3.5 Online kompilátor

Nyní se dostáváme k tomu, jak jsem postupoval při realizaci online kompilátoru. V závěrečném rozhodování jsem měl na výběr mezi využitím API, integrování celého (embedding) existujícího kompilátoru, nebo navrhnutí vlastního kompilátoru. Jelikož jsem vytvářel aplikaci mířenou pro začátečníky, rozhodl jsem se o navrhnutí vlastního kompilátoru, který by využíval ukládání napsaného kódu do souborů a následné kompilaci pomocí balíčků jako například Java Development Kit pro kompilaci souborů s příponou .java.



Obr. 26: Vzhled kompilátoru
(vlastní zpracování)

Než jsem se pustil do samotného kompilování napsaných programů, musel jsem přijít na to, jak udělat, aby pole pro psaní vypadalo jako běžné prostředí, se kterým se může člověk při programování setkat. V tomto kroku jsem využil javascriptovou knihovnu Ace, která poskytuje základní vzhled a zvýrazňování syntaxe kódu. Díky této knihovně jsem byl schopen vyřešit pole, kam uživatel bude psát vlastní kód pro vyzkoušení. Aby používaná knihovna fungovala správně, je nutné hlídat jazyk, který si uživatel vybere při programování a podle toho měnit mód, ve kterém Ace funguje, k tomu jsem využil jednoduchou funkci, která nastaví režim podle vybraného jazyka z nabídky. Kromě přepnutí režimu se také textové pole naplní základním kódem, který vypíše na obrazovku Hello World.

```

function setEditorModeAndCode(pjazyk) {
    if (pjazyk === "c") {
        editor.session.setMode("ace/mode/c_cpp");
    } else if (pjazyk === "cpp") {
        editor.session.setMode("ace/mode/c_cpp");
    } else if (pjazyk === "java") {
        editor.session.setMode("ace/mode/java");
    }

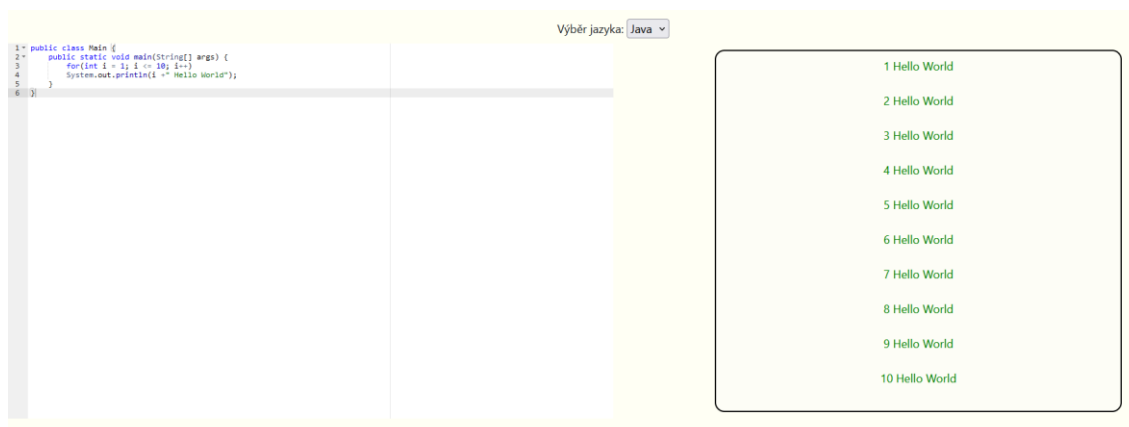
    if (pjazyk === "c") {
        editor.setValue("#include <stdio.h>\n\nint main() {\n    printf(\"Hello World\\n\");\n    return 0;\n}");
    } else if (pjazyk === "cpp") {
        editor.setValue("#include <iostream>\n\nint main() {\n    std::cout << \"Hello World\\n\";\n    return 0;\n}");
    } else if (pjazyk === "java") {
        editor.setValue("public class Main {\n    public static void main(String[] args) {\n        System.out.println(\"Hello World\\n\");\n    }\n}");
    }
    editor.gotoLine(1);
}

```

Obr. 27: Ukázka přepínání jazyků
(vlastní zpracování)

Po dokončení toho, jak by měl kompilátor vypadat začala práce na samotné kompilaci textu, který se může objevit v editoru. Prvním a tím nejjednodušším krokem bylo dostat obsah z editoru do souboru, který by se později dal zkompileovat. Pro vytvoření souboru jsem opět využil ajaxu, kde jsem posílal celkem tři hodnoty, první hodnotou byl zvolený jazyk, druhou hodnotou byl obsah editoru a poslední hodnotou uživatelské jméno. Na základě těchto informací se vytvořil soubor, do kterého se zapsal obsah editoru a pro příponu souboru se využilo hodnoty ze zvoleného jazyku, soubory se vytvářely v nově vytvořené složce, která nesla jméno uživatele. Už v tomto kroku jsem myslel na možné zneužití konzole a vyhledával možné příkazy, které by mohly narušit funkci jak samotného kompilátoru, tak celé aplikace. Skončil jsem tedy s dlouhým polem, které obsahuje přes čtyřicet příkazů ve spojení s regulárními výrazy, abych pokryl mezery mezi textem. Pokud se v editoru objeví jakýkoliv z těchto příkazů a uživatel se ho bude pokoušet spustit, automaticky se vypíše zpráva o nezdařené kompilaci. Dalším opatřením bylo stanovení maximální doby, kterou může skript běžet na pět vteřin pomocí funkce `set_time_limit(5);`. Jako další opatření sloužilo nastavení tlačítka, které zajistí, že tlačítko nepůjde zmáčkнут, dokud se na výstupu neobjeví jakákoliv hodnota, ať už se jedná o chybovou hlášku nebo samotný výstup programu.

Jako první jazyk, který jsem se pokusil zprovoznit byl jazyk Java, jak už bylo zmíněno využil jsem balíček `java development kit`, který obsahuje potřebné nástroje pro kompilaci a spouštění aplikací. Jako první jsem si definoval proměnnou, která obsahuje cestu k balíčku `jdk`, dále se vytvořil soubor `Main.java`, do kterého se zapsal obdržенý kód z editoru. Pro vytvoření, zápis a zavření souboru jsem využil php funkcí `fopen()`, `fwrite()` a `fclose()`. Jako další následuje kompilace, při které se vytvoří soubor s příponou `.class`. Tato akce se provede využitím příkazu `shell_exec()`. V tento moment se kontroluje úspěšná kompilace a případné odeslání chybové hlášky do výstupní konzole. Pokud kompilace kódu proběhne bez chyb, dojde ke spuštění programu a uložení výstupu. Před odesláním výsledku do výstupní konzole se přepíše zobrazení nových řádků pomocí funkce `str_replace()`.



Obr. 28: Ukázka jednoduchého programu
(vlastní zpracování)

Po dokončení části kompilátoru, který by byl schopný zpracovávat jednoduché Java programy následovalo řešení pro jazyky C a C++.

Pro řešení jsem využíval podobného principu. Jako první jsem si stáhnul balíček souboru, který obsahuje potřebné nástroje pro kompilaci, v těchto případech se jednalo o nástroje jako gcc.exe a g++.exe, které umožňují kompilovat zdrojový kód z jazyků C a C++ do spustitelných souborů. Postup byl tedy následující, zdrojový kód z editoru se zapíše do souboru, který je potom zkompileován a pokud se nevyskytnou žádné chyby při kompilaci vzniká nový .exe soubor, který je spuštěn a výstup programu je uložen do proměnné, která se posílá do výstupní konzole pro uživatele. Toto řešení fungovalo, ale mělo své problémy, které se pokusím shrnout v závěru práce.

3.6 Zabezpečení aplikace

Co se týče zabezpečení aplikace, tak už od začátku jsem se snažil dávat velký pozor na to, co jsou uživatelé schopní odesílat za data ve formulářích. V jednotlivých polích využívám kontrolu znaků, které by mohly jakýmkoliv způsobem manipulovat s daty v databázi, nebo při samotném odesílání dat využívám připravené dotazy, které jsou součástí mysql a pomáhají v prevenci sql injection. Kromě prevence sql injection jsem si dal také pozor na cross-site scripting, pro tyto případy jsem využil php hlaviček, které omezují vkládání obsahu na webové stránky. Dalším opatřením byla dostupnost kompilátoru, z počátku jsem myslel, že bude v pořádku poskytnout přístup k online kompilátoru všem návštěvníkům, ovšem po průběžném testování jsem se rozhodl pro omezení jenom pro přihlášené uživatele. Dalším opatřením bylo využití .htaccess pro skrytí celých cest k jednotlivým souborům a přesměrování na chybovou stránku, pokud se uživatel pokusí vyhledat stránku, která neexistuje.

Závěr

Cílem bakalářské práce bylo vytvořit webovou aplikaci, která pomůže jednotlivcům dostat se do světa programování. Uživatelé mají možnost naučit se základy programování v jazycích C, C++ a Java, společně se základy tvoření statických webových stránek s využitím HTML a CSS. Aplikace byla během vývoje průběžně testována a řekl bych, že cílů se podařilo dosáhnout, z nabitých zkušeností ovšem můžu říci, že pokud bych se znovu pokoušel o vytváření podobné aplikace, zvolil bych jiná přístup a jiná metody.

Teoretická část se zabývala využitými technologiemi, které obsahovaly názorné ukázky a už v této době se dá říct, že jsem se snažil představit použitelné technologie a jak jednoduché je s nimi pracovat.

Největší podíl při vývoji měl jazyk PHP. Znalosti PHP jsem si díky této práci prohloubil, jelikož jsem se často s týkal s novými věcmi, hlavně při vytváření kompilátoru.

Během vývoje jsem se občas potkal s problémy, které částečně vznikly kvůli tomu, že jsem se snažil práci řešit co nejjednodušším způsobem a nevyužívat tak všech možných nástrojů, jen proto, abych mohl po dokončení práce na stránkách uvést, že k vytvoření rozsáhlé webové aplikace stačí málo a začátečníci by se neměli bát experimentovat.

Práce by se dala nadále vylepšit, jedním z aspektů, které by se určitě daly vylepšit je přidávání lekcí do jednotlivých programovacích jazyků, aktuálně se každá lekce musí přidávat ručně, práci bych rozšířil jednoduchým systémem, který by se staral o správu lekcí, kde by se lekce mohly jednoduše přidávat, upravovat a mazat. V práci bych dále změnil přístup, jakým se řeší realizace kompilátoru, jelikož jsem aplikaci průběžně testoval lokálně neměl jsem tak často příležitost se potýkat s problémy, se kterými jsem se setkal, jakmile jsem měl možnost aplikaci nahrát na server. Protože jsem se odvážil realizovat kompilátor podle sebe, narazil jsem na problémy, se kterými bych se nesetkal, kdybych využil alternativní možnost vypracování, o kterých jsem se rozhodl před začátkem realizace kompilátoru. Pro práci bych také zvolil přístup pomocí jednoho z dostupných php frameworků jako je například symfony, nebo Nette.

Seznam použité literatury

Bringing MySQL to the web. <https://www.phpmyadmin.net/> [online]. [cit. 2023-01-14]. Dostupné z: <https://www.phpmyadmin.net/>

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

ČÁPKA, David. Lekce 6 - Úvod do CSS (kaskádových stylů). *ITnetwork* [online]. [cit. 2021-03-28]. Dostupné z: <https://www.itnetwork.cz/html-css/webove-stranky/jak-psat-moderni-web-html-tutorial-uvod-do-css>

JANOVSKÝ, Dušan. PHP -- Jak začít. *Jak psát web* [online]. [cit. 2021-03-28]. Dostupné z: <https://www.jakpsatweb.cz/php/jak-zacit.html>

KOĐOUSKOVÁ, Barbora. JavaScript pro začátečníky: co to je a jak funguje. *Rascasone.com* [online]. 2022 [cit. 2023-01-15]. Dostupné z: <https://www.rascasone.com/cs/blog/co-je-javascript-pro-zacatecniky>

Lekce 1 - Úvod do CSS frameworku Bootstrap Zdroj: <https://www.itnetwork.cz/html-css/bootstrap/kurz/uvod-do-css-frameworku-bootstrap>. *Itnetwork.cz* [online]. [cit. 2023-01-15]. Dostupné z: <https://www.itnetwork.cz/html-css/bootstrap/kurz/uvod-do-css-frameworku-bootstrap>

Lekce 4 - Debugging ve vývojovém prostředí Visual Studio Code. *Itnetwork.cz* [online]. [cit. 2022-11-25]. Dostupné z: <https://www.itnetwork.cz/javascript/debugging-vscode-webstorm/debugging-ve-vyvojovem-prostredi-visual-studio-code>

PHP. https://wikisofia.cz/wiki/Hlavn%C3%AD_strana [online]. [cit. 2023-01-13]. Dostupné z: <https://wikisofia.cz/wiki/PHP>

StarUML. <https://staruml.io/> [online]. [cit. 2023-01-12]. Dostupné z: <https://staruml.io/>

Přílohy

Přílohou této práce je flash disk, který obsahuje složku s projektem, součástí projektu je manuál, kde je možné najít základní informace a sql dotazy, které umožňují vytvoření databáze a vložení testovacích dat. Uložiště také obsahuje tuto práci ve formátu PDF.