Pracovní listy pro výuku webových technologií na střední škole

Bc. Vojtěch Reiner

Diplomová práce
2020

Univerzita Tomáše Bati ve Zlíně
Fakulta aplikované informatiky

Univerzita Tomáše Bati ve Zlíně

Fakulta aplikované informatiky

Ústav informatiky a umělé inteligence

Akademický rok: 2019/2020

ZADÁNÍ DIPLOMOVÉ PRÁCE

(projektu, uměleckého díla, uměleckého výkonu)

Jméno a příjmení:	**Bc. Vojtěch Reiner**
Osobní číslo:	**A18414**
Studijní program:	**N3902 Inženýrská informatika**
Studijní obor:	**Učitelství informatiky pro střední školy**
Forma studia:	**Prezenční**
Téma práce:	**Pracovní listy pro výuku webových technologií na střední škole**
Téma práce anglicky:	**Worksheets for Teaching Web Technologies in High School**

Zásady pro vypracování

1. Proveďte literární rešerši na téma www technologií a projektové výuky.
2. Formou projektu navrhněte vhodné oblasti učiva pro výuku webových technologií.
3. Realizujte zvolené oblasti jako souhrn systémově řešených úloh.
4. Získejte zpětnou vazbu od studentů střední školy.
5. Zhodnoťte výsledky práce a její realizace.

Rozsah diplomové práce:
Rozsah příloh:
Forma zpracování diplomové práce: **tištěná/elektronická**

Seznam doporučené literatury:

1. DVOŘÁČEK, J. Pedagogika pro učitele odborných předmětů, 2005, VŠE, ISBN: 80-245-0886-9.
2. DVOŘÁKOVÁ, Markéta. Projektové vyučování v české škole: vývoj, inspirace, současné problémy. Praha: Karolinum, 2009. ISBN 978-80-246-1620-9
3. GILMORE, W. J. *Velká kniha PHP 5 a MySQL: kompendium znalostí pro začátečníky i profesionály*. Nové, 3. vyd. Přeložil Jan POKORNÝ. Brno: Zoner Press, 2011. Encyklopedie Zoner Press. ISBN 978-80-7413-163-9.
4. KOSEK, Jiří. *PHP a XML*. Praha: Grada, 2009. Profesionál. ISBN 978-80-247-1116-4.
5. KRATOCHVÍLOVÁ, Jana. Teorie a praxe projektové výuky. Brno: Masarykova univerzita, 2006. ISBN 80-210-4142-0

Vedoucí diplomové práce: **prof. Mgr. Roman Jašek, Ph.D.**
Ústav informatiky a umělé inteligence

Datum zadání diplomové práce: **28. listopadu 2019**
Termín odevzdání diplomové práce: **15. května 2020**

doc. Mgr. Milan Adámek, Ph.D.
děkan

prof. Mgr. Roman Jašek, Ph.D.
ředitel ústavu

Ve Zlíně dne 9. prosince 2019

Prohlašuji, že

- beru na vědomí, že odevzdáním diplomové práce souhlasím se zveřejněním své práce podle zákona č. 111/1998 Sb. o vysokých školách a o změně a doplnění dalších zákonů (zákon o vysokých školách), ve znění pozdějších právních předpisů, bez ohledu na výsledek obhajoby;
- beru na vědomí, že diplomová práce bude uložena v elektronické podobě v univerzitním informačním systému dostupná k prezenčnímu nahlédnutí, že jeden výtisk diplomové/bakalářské práce bude uložen v příruční knihovně Fakulty aplikované informatiky Univerzity Tomáše Bati ve Zlíně a jeden výtisk bude uložen u vedoucího práce;
- byl/a jsem seznámen/a s tím, že na moji diplomovou práci se plně vztahuje zákon č. 121/2000 Sb. o právu autorském, o právech souvisejících s právem autorským a o změně některých zákonů (autorský zákon) ve znění pozdějších právních předpisů, zejm. § 35 odst. 3;
- beru na vědomí, že podle § 60 odst. 1 autorského zákona má UTB ve Zlíně právo na uzavření licenční smlouvy o užití školního díla v rozsahu § 12 odst. 4 autorského zákona;
- beru na vědomí, že podle § 60 odst. 2 a 3 autorského zákona mohu užít své dílo – diplomovou práci nebo poskytnout licenci k jejímu využití jen připouští-li tak licenční smlouva uzavřená mezi mnou a Univerzitou Tomáše Bati ve Zlíně s tím, že vyrovnání případného přiměřeného příspěvku na úhradu nákladů, které byly Univerzitou Tomáše Bati ve Zlíně na vytvoření díla vynaloženy (až do jejich skutečné výše) bude rovněž předmětem této licenční smlouvy;
- beru na vědomí, že pokud bylo k vypracování diplomové práce využito softwaru poskytnutého Univerzitou Tomáše Bati ve Zlíně nebo jinými subjekty pouze ke studijním a výzkumným účelům (tedy pouze k nekomerčnímu využití), nelze výsledky diplomové práce využít ke komerčním účelům;
- beru na vědomí, že pokud je výstupem diplomové práce jakýkoliv softwarový produkt, považují se za součást práce rovněž i zdrojové kódy, popř. soubory, ze kterých se projekt skládá. Neodevzdání této součásti může být důvodem k neobhájení práce.

Prohlašuji,

- že jsem na diplomové práci pracoval samostatně a použitou literaturu jsem citoval. V případě publikace výsledků budu uveden jako spoluautor.
- že odevzdaná verze diplomové práce a verze elektronická nahraná do IS/STAG jsou totožné.

Ve Zlíně, dne 30.7.2020

Vojtěch Reiner, v.r.
podpis diplomanta

ABSTRAKT

Ústředním tématem diplomové práce je výuka WWW technologií na střední škole. Práce se v pedagogické části zabývá rozborem forem výuky, projektovou výukou, zásadami správného vyučování a RVP související s tvorbou webových stránek. Technická část řeší bezpečnost na internetu z uživatelského pohledu, tvorbu webové stránky z hlediska šablony (frontend), programování webových aplikací, rozebírá dostupné možnosti tvorby webu a používaný software. V závěru jsou řešeny také databáze, jako hlavní úložiště dat pro WWW stránky. V praktické části se práce zabývá návrhem projektové výuky, jejíž součástí je 16 lekcí pro výuku tvorby WWW stránek a 6 projektových úkolů. Navržené pracovní listy a podklady k výuce jsou nedílnou součástí této práce jako příloha. Kurz sjednocuje úroveň studentů ve znalostech ohledně používání internetu a bezpečnosti na internetu. Následně se zabývá tvorbou webových stránek, algoritmizací a programováním aplikací a systémů pro webové stránky. Součástí je také zp. vazba od studentů SŠ z realizované výuky.

Klíčová slova: projektová výuka, webové stránky, tvorba WWW stránek, bezpečnost na internetu, zabezpečení webu, programování webových aplikací, programování v PHP, tvorba HTML šablony

ABSTRACT

The main theme of the thesis is teaching WWW technologies at secondary school. The thesis in the pedagogical part describes the analysis of forms of teaching, project teaching, principles of correct teaching and RVP about creating websites. The technical part solves internet security from the user's point of view, creating a website from the point of view of a template (frontend), programming web applications, analyzes the available possibilities of creating a website and software for it. In the end of thesis, are descriped database systems. Practical part is about teaching WWW technologies at secondary school. In attachment is 16 lessons and 6 projects tasks for teaching the creation of web pages, which are an integral part of the thesis. The course unifies students' level of knowledge about Internet usage and Internet security. Then it continume by creating web pages, algorithms and programming applications and systems for creating websites. It also includes feedback from secondary school students from realized lessons.

Keywords: project education, web pages, web pages creation, internet security, web security, web application programming, PHP programming, HTML template creation

„Pokud se očekává, že stroj bude neomylný, nemůže být také inteligentní.“

Alan Turing

Děkuji vedoucímu práce prof. Romanu Jaškovi za vedení a odborné připomínky k diplomové práci.

Prohlašuji, že odevzdaná verze bakalářské/diplomové práce a verze elektronická nahraná do IS/STAG jsou totožné.

OBSAH

ÚVOD

Ačkoliv jsou Internet a s ním úzce spojené webové technologie poměrně novodobou záležitostí, v ČR používaných širokou veřejností cca dvě desítky let, jsou nedílnou součástí každodenního života každého občana České republiky. Výuka WWW technologií je zařazována pouze velmi okrajově v několika málo hodinách, případně vůbec. Celý předmět se této tématice věnuje pouze na odborných školách zaměřených na informatiku. Moje diplomová práce se zabývá přípravou podkladů pro projektovou výuku WWW technologií na střední škole. V teoretické části rozeberu používané technologie a vše související s tvorbou webu, včetně rozboru výukových metod.

V praktické části práce je řešeno vytvoření výukových lekcí + pracovních listů pro projektovou výuku webových technologií na střední škole. Kurz je koncipován pro odborné školy s rozšířenou výukou informatiky, ale jeho dílčí části (např. bezpečnost na internetu, nebo základy tvorby www, algoritmizace) mohou být využity na kterékoliv střední škole. Internet studenti používají denně a je důležité mít alespoň základní povědomí o jeho fungování a především bezpečnosti. Pracovní listy tak využijí i ostatní střední školy, tak jak jim využití dovolí časová dotace a ŠVP. V rámci kurzu je řešena také algoritmizace, která je nedílnou součástí programování a kurz tak lze využít, případně upravit pro výuku jiného jazyka, než je v kurzu navrhnuté PHP.

I. TEORETICKÁ ČÁST

1 FORMY VÝUKY

Pojem forma výuky, nebo také organizační forma výuky znamená organizaci vyučovací hodiny, tedy jakým způsobem výuka probíhá. Každá forma výuky rozvíjí různé kompetence studentů a některé jsou pro určité předměty vyloženě typické.

1.1 Frontální výuka

Frotnální výuka, označovaná také jako hromadná je jednou z prvních a i v současné době nejrozšířenějších forem výuky. Třída je v tomto případě celek a učitel na studenty působí hromadně, všichni mají stejné úkoly ve stejném čase. Studenti sedí dle zasedacího pořádku a učitel je v popředí, kde vede přednášku, doplňuje svoji prezentaci atp. Výhoda této výukové formy tkví v tom, že je možné pracovat s velkým množstvím studentů najednou, čili přípravu má vyučující pouze jednu. Důležité je aby byla skupina (třída) na stejné vzdělávací úrovni. Tempo je přizpůsobeno průměrným žákům.

Frontální výuka představuje „tradiční způsob vyučování, v němž učitel pracuje hromadně se všemi žáky ve třídě jednou společnou formou, se stejným obsahem činnosti“.

[1]

1.2 Individuální výuka

Tato forma výuky není typická pro školní prostředí, dochází zde k interakci jednoho učitele a žáka. Jak již z názvu vyplývá, učitel může využít individuálního přístupu, zaměřit se na konkrétní problémy žáka a naopak některé pasáže probrat rychleji, zjistí-li, že je žák dostatečně pochopil. S individuální výukou se setkáme zejména při doučování, soukromých hodinách, domácím vyučování, v rámci umělecké výchovy a dalších.

1.3 Skupinová a kooperativní výuka

U skupinové výuky jsou studenti rozděleni do menších skupin, v rámci kterých probíhá spolupráce při řešení konkrétního úkolu. Počet studentů ve skupině je individuální, obvykle mezi 4 – 6 studenty. Učitel by měl kontrolovat zapojení všech členů a případně řešit nevyváženost některých skupin. Skupiny můžou být vytvořeny vyučujícím – tzv. formální vytvoření skupin, nebo samotnými studenty – neformální vytvoření skupin. Můžeme tvořit homogenní skupiny, např. žáky s podobným prospěchem, kdy dle jejich úrovně zadáváme

různě obtížné úkoly. Nechceme-li žáky diferencovat, vytvoříme heterogenní skupiny, nezávislé na prospěchu, zálibách, pohlaví atp. Organizace skupiny:

- Kruhová – všichni studenti v rámci skupiny pracují na stejném zadání
- Paralelní – každý člen skupiny řeší nějaký pod-úkol a vyřešením všech diskrétních úkolů získáme finální řešení
- Hvězdicová – u tohoto typu organizace je určen vedoucí skupiny, který studentům rozděluje práci a řídí pracovní postup

Kooperativní výuka je velmi podobná skupinové, zásadní rozdíl je ve větším důrazu na jednotlivce. Při kooperativní výuce má každý člen více vlastní zodpovědnosti. Mezi nejdůležitější vlastnosti kooperativní výuky patří sdílení, komunikace a spolupráce. Dle statistik je při kooperaci nejvýhodnější volit čtyřčlenné skupiny.

1.4 Projektová výuka

Projektová výuka bývá mezi odbornou veřejností zařazována do výukových forem (Kašová, 1995, Dvořáková 2009), ale i výukových metod (Maňák 2003), či výukových strategií. Realizace projektové výuky je poměrně novodobou záležitostí, reagující na potřeby získávání kompetencí v oblasti řešení problémů a propojením s praktickými úkoly. Obvykle je projektová výuka založena na řešení konkrétního problému, který musí být promyšlený, dostatečně zajímavý a propracovaný. Projekty podporují kooperaci a zároveň motivují studenty k obvykle samostatné činnosti, výsledkem může být i vytvoření nějakého produktu. Může být skupinový, i individuální. Vede studenty k vlastní zodpovědnosti za svou práci a učí je pracovat s různými zdroji informací. Výstupem projektu, resp. projektového úkolu je vyřešení nějakého problému, zhotovení programu, výrobku atp. Znaky projektu:

- Klíčovou roli v projektu hraje sám student
- Student je zodpovědný za svou práci, učí se samostatnosti, ale i spolupráci s ostatními
- Dává příležitost studentovi organizovat si čas
- Dává příležitost skupině studentů rozdělit si role v projektu
- Relevantnost pro praktické využití
- Zřetel k aktuální situaci (aktuálnost tématu)

Projekt je komplexní úkol (problém), který je spjatý s životní realitou, s nímž se žák identifikuje a přebírá za něj zodpovědnost, aby svou teoretickou i praktickou činností dosáhl výsledného žádoucího produktu (výstupu) projektu, pro jehož obhajobu a hodnocení má argumenty, které vycházejí z nově získané zkušenosti

[2]

Projektová výuka si našla své místo v odborných a praktických předmětech, kde lze nalézt dostatek témat pro realizaci výuky. Např. v programování je každý program určitý problém, který musíme nějak vyřešit. Ať tedy programujeme kalkulačku, nebo systém banky, jedná se o problém, který lze řešit formou projektu a projektové výuky. Stejně je tomu tak v pracovních činnostech, kdy mají žáci za úkol vytvořit nějaký výrobek. Problém mohou konzultovat se spolužáky (dle zadání projektu), dohledávat si potřebné informace v literatuře, hodnotit výrobky ostatních spolužáků atd.

Definice pánů Maňáka a Švece říká, že *projektová výuka částečně navazuje na metodu řešení problémů, jde však v ní o problémové úlohy komplexnější, výukové záměry a plány, které mají vždy také širší praktický dosah. Jestliže výuka zaměřená na řešení učebních problémů a úloh se uzavírá mezi stěny učebny, nebo školy, učení v projektech hranice školy překračuje. Tradiční výuka také většinou probíhá v izolovaných vyučovacích předmětech, kdežto projekty sdružují přirozenou cestou k spolupráci několik vyučovacích předmětů, neboť jejich cílem je řešit situaci* ***životní reality****.*

Projekty dělí podle časových rozsahů na:

- Krátkodobé (v řádu hodin)
- Střednědobé (1-2 dny)
- Dlouhodobý (projektový týden)
- Mimořádně dlouhodobý (týdny i měsíce)

Z hlediska práce na projektu je lze rozdělit na:

1) Stanovení cíle (s ohledem na vhodnost, realizovatelnost a motivaci žáků)
2) Vytvoření plánu řešení (definování podmínek, zajištění odpovědnosti za splnění jednotlivých úkolů, prodiskutování plánu se studenty a výběr úkolů pro každého studenta/skupinu studentů)

3) Realizace plánu (vedoucí projektu v průběhu stav práce kriticky sleduje a srovnává plán s aktuálním stavem. Studenti realizují aktivity, které mají zajistit úspěšné splnění – vyhledávání informací, zajišťování materiálu, provádění měření atd. Studenti se učí experimentovat, učí se zodpovědnému jednání, využívají různé informační zdroje atd.)
4) Vyhodnocení (opírá se o sebekritiku a objektivní posouzení přínosu jednotlivých řešitelů – v případě skupinového projektu. Následuje zveřejnění výsledků a celkové zhodnocení práce na projektu. S výstupem můžeme seznámit školní, nebo i širší veřejnost, což má značný motivační vliv na řešitele, posiluje jeho sbedůvěru, zejména u slabších studentů, což se v tradiční výuce často nedostavuje)

[3]

2 ZÁSADY SPRÁVNÉ VÝUKY

Každý, kdo se zabývá pedagogikou musí bezpodmínečně znát Jana Amose Komenského, průkopníka v oblasti školství, autora výroku „škola hrou“ a mnoha zajímavých literálních děl, souvisejících nejen z výchovou a vzděláváním. Protože tento autor k pedagogice a výuce neodmyslitelně patří, dovolím si citovat několik odstavců z jeho didaktiky analytické.

- *Ukázka nechť vždy kráčí vpředu, poučka ať vždy následuje, napodobení nechť je vždy důrazně požadováno. Ujal se zvyk, že se jednotlivým oborům vyučuje poučkami a teprve potom se osvětluje smysl pravidel příklady. Ale přirozenější je pořadí, kde příklady předcházejí.*
- *Všude je třeba začít od nečetných, krátkých, jednoduchých, obecných, blízkých, pravidelných a ponenáhlu postupovat k četnějšímu, k věcem obšírnějším, složenějším, zvláštnějším, odlehlejším a nepravidelným.*
- *Věci, které postrádají souvislosti, mohou být stěží chápány a posuzovány a tak i stěží svěřovány paměti.*
- *Vždy postupně, nikdy skokem*
- *Nesváděj své vlastní neúspěchy na nepořádky v dodavatelských vztazích. Upřesnění: Nesvádějte nesprávnou výuku na „hloupé“ děti ze ZŠ, domnívajíc se, že výuku nezvládli naši kolegové na předchozím stupni vzdělávání.*

[4]

Z posledního odstavce je možné vyvodit a v praxi se tak opravdu děje, že někteří vyučující jsou nespokojeni s, řekněme studijní úrovní studenta (tedy získanými znalostmi a kompetencemi) a tento fakt přisuzují jiným vyučujícím. Bude-li tato domněnka oprávněná, je možné situaci s daným učitelem konzultovat a dobrat se kompromisu ve zjištěných problémech. Nicméně výuku to nesmí za žádných okolností ovlivnit a je naprosto nevhodné své kolegy takto před studenty dehonestovat.

Dále uvádím metodické zásady, nasbírané během magisterského studia (majoritní část uváděl doc. Botek).

- Nemůžeš-li nic naučit, snaž se studenty alespoň neotrávit.
- Každý student by měl odcházet minimálně s takovou chutí, s jakou přišel.
- Pamatuj, že první mohou být poslední a poslední prvními.

- Cílem tvých přednášek není, abys vypadal jako velký odborník, ale aby se stali odborníci z Tvých posluchačů.
- Pokud to neumíš vysvětlit jednoduše, nerozumíš tomu dostatečně dobře.
- Zamýšlej se nad tím, co jim sděluješ.
- Nikdy si nepamatujte něco, co můžete vyhledat.
- Měl bys učit studenty programovat lépe, než-li to umíš Ty.
- Příklady by měli být zajímavé a složitější, aby se museli zamyslet a nastudovat si je.
- Informatika není exaktní věda jako matematika, nejsou to axiomy, ale doporučení co by se mělo (téměř jako morální zásady)
- Tvrzení, jehož negace je nesmysl je buď trivialita, nebo také nesmysl.
- Neexistuje jiná rozumná forma výchova než být příkladem - nelze-li jinak, tedy odstrašujícím.
- Ovládání programovacího jazyka a počítače je pouze prostředek k našemu cíli – důležitá je schopnost řešit problémy.
- Nejproduktivnější formou učení informatiky je opravování studentských prací před studenty a diskuse nad nimi.
- Dobrý program má vlastnosti: správnost, čitelnost, modifikovatelnost a efektivita.
- Pět let svého působení na škole jsem si myslel, že odbornost je to nejdůležitější. Druhých pět let jsem se domníval, že mé pedagogické znalosti a dovednosti jsou to nejdůležitější.
  Po 10. letech jsem zjistil, že nejdůležitější je slušnost a schopnost jednat s dětmi.

2.1 Využití prekonceptů

Je nutné vědět, na jaké znalosti studentů můžeme navázat, zda se již s daným tématem, alespoň okrajově setkali a mají nějaké základní znalosti (pre-koncepty), nebo naopak o tématu nic neví, případně mají mylné představy (mis-koncepty). U WWW technologií je předpoklad, že se s nimi každý student SŠ setkal a v mnoha tématech můžeme navázat na používání Internetu na denní bázi, prohlížení webů, zadávání hesel, obecně bezpečností

Internetu atp. Důležité je zjistit tyto informace, např. diskusí, případně brainstormingem na začátku hodiny, ale i pomocí jiných metod.

Žák přichází do školy nikoli jako „tabula rasa“, kterou teprve učitel popíše svým snažením, nýbrž si přináší své dětské představy o obsahu pojmů, své názory na svět, má své předivo vztahů mezi pojmy. Z pohledu školy jde však o představy a vztahy, které vznikly nahodile, nesystematicky, nevědecky. Škola (pokud vůbec o žákovských představách uvažuje) je podceňuje či dokonce ignoruje, což je to nejhorší, co může udělat. Existují ovšem výjimky, jak mezi učiteli, tak mezi výzkumnými pracovníky.

[5]

Proto je důležité, aby učitel s prekoncepty pracoval, snažil se na ně navázat a podnítit studenty k přemýšlení. Zejména v programování a obecně v algoritmizaci, je velmi důležitý vlastní úsudek a správných cest, jak dojít k cíli je obvykle více. Jako praktický příklad pánové Mareš a Ouhrabka uvádí:

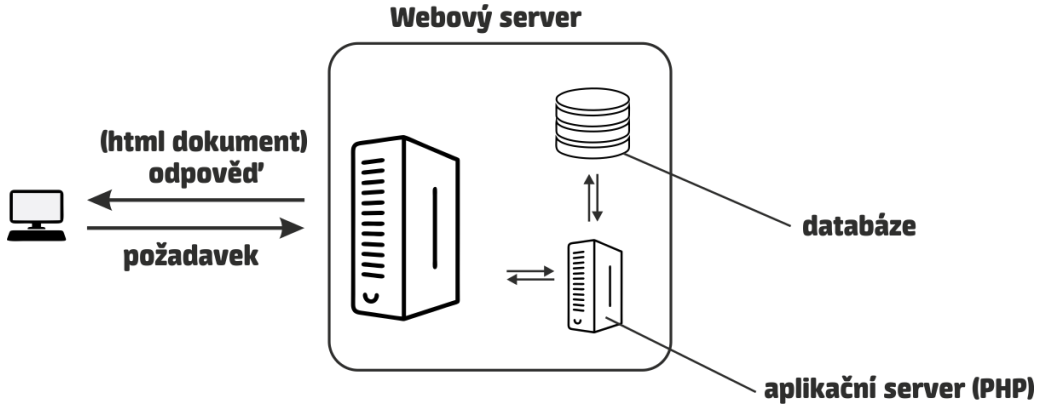
Žák přichází do 3. ročníku základní školy jazykově připraven, intuitivně „rozumí“ souslovím: silný člověk, silný strom, silně vojsko, silný vítr. Avšak učitel přírodovědy musí tuto prekoncepci(vázanou na řadu představ / pohádek, dětské četby, či na žákovu vlastní zkušenost) zcela přeměnit. Učebnice má jediné slabé místo, a tím je právě kapitola o síle. Začíná výrazem „mít sílu“, a tak podporuje zcela nevhodně jednu z častých prekoncepcí síly, žák se později těžko zbavuje snahy přivlastňovat sílu tělesům. Přitom síla, jako fyzikální veličina rozhodně není vlastností těles. Učitele, zejména učitele fyziky, chemie, biologie lze zjednodušeně rozdělit do dvou skupin. Učitelé zaměření prakticky, zaměření na operacionalizaci poznatků, přimějí žáka, aby si osvojil minimální, ale konzistentní a použitelný soubor poznatků a představ. Tím (zpravidla nevědomě) vytěsňují z žákova myšlení mylně představy nebo zabraňují, aby chybné představy u žáků vznikly. Naopak učitelé, kteří jsou zaměřeni dogmaticky, sdělují a zkoušejí předepsané učivo a nezajímá je, co si žák pod učivem sám představuje.

[5]

3 WWW TECHNOLOGIE

3.1 Webový server

Pojem webový server se mezi laickou veřejností ujal zejména v souvislosti s webovou stránkou. Uživatelé tedy obvykle myslí webovou stránku, chtějí-li sdělit, že navštívili nějaký webový server. Ve skutečnosti tento pojem značí počítač, díky kterému se na onu webovou stránku dostaneme.



Obrázek 1 – Schéma webového serveru

Jak je možné vidět na schématu výše, webový server přijímá požadavek uživatele na konkrétní webovou stránku (zadáním URL do prohlížeče), ten jej zpracuje a odesílá zase zpět. Přenos probíhá pomocí protokolu HTTP(S). Zpracování může být jednoduché zaslání statické webové stránky, nebo složitější zpracování dynamického obsahu, ukládání, nebo naopak získávání obsahu z databáze atp. Základní webový server nám dokáže stránku pouze zobrazit. Pro práci s databází a dynamickými scripty je třeba nainstalovat i komponenty pro jejich zpracování.

Apache HTTP Server je softwarový webový server s otevřeným kódem pro GNU/Linux, BSD, Solaris, Mac OS X, Microsoft Windows a další platformy. V současné době dodává prohlížečům na celém světě většinu internetových stránek.

Pokud se rozhodnete na svém serveru provozovat webové stránky, bude Apache logická volba. Apache samotný slouží pro zobrazování stránek a pokud potřebujete server na složitější prezentaci, například CMS systémy WordPress a Drupal, potřebujete i podporu PHP a databázi. Tyto nejpoužívanější komponenty se označují jako LAMP server podle počátečních písmen Linux, Apache, MySQL a PHP.

[6]

Důležité je zmínit, že k zobrazení statické webové stránky nepotřebujeme žádný webový server, to má na starosti prohlížeč. Mezi nejpoužívanější prohlížeče aktuálně patří Mozila Firefox, Google Chrome, Internet Explorer.

Prohlížeč je v podstatě takový interpreter, který nám převede šablonu (HTML kód) na graficky přijatelnou webovou stránku. To, jakým způsobem to udělá, je závislé na jeho vnitřním algoritmu a proto se může stát, že se stejná webová stránka zobrazí v každém prohlížeči trochu jinak. Takový web je třeba odladit tak, aby se zobrazoval správně (stejně) ve všech nejpoužívanějších prohlížečích. Stejně tak je důležité web přizpůsobit různým zařízením. Dnes už si webové stránky nezobrazují pouze uživatelé stolních počítačů s víceméně jednotným rozlišením, ale nedílnou část uživatelské základny tvoří notebooky, tablety a mobilní telefony.

Ke zprovoznění PHP scriptů, tedy dynamických webových stránek ale budeme potřebovat webový server, resp. jeho část, která se stará právě o jejich zpracování. Nejinak je tomu u databází. My si takový server můžeme zprovoznit na vlastním počítači a námi vytvořenou stránku si tak odladit offline. V době, kdy bylo připojení k internetu pomalé, nestabilní a finančně náročné to byla jediná možnost. Postupem času se ale stává méně oblíbenou a já osobně jsem ji ve svém profesním životě nevyužil už řadu let. Mnohem zajímavější je druhá možnost, využití serveru někoho jiného.

Zmíněný server je k dispozici v rámci tzv. webhostingu, kterých je dnes celá řada a k dispozici jsou některé zcela zdarma, jiné v řádu desetikorun za měsíc. Výhoda je v tom, že máme k dispozici pravidelné zálohy, náš script, resp. webovou stránku si můžeme zobrazit odkudkoliv kde je připojení k internetu a také z jakéhokoliv zařízení. Založení není nic složitého a vytvořit si databázi zabere také pár minut. K serveru je možné se připojit pomocí FTP klienta (např. Total Commander, Filezila a další), nebo přes webovou aplikaci poskytovatele webhostingu (méně pohodlné).

Mimo výše zmíněného Apache existují i jiné systémy pro webové servery, např. NGINX, LiteSpeed a další.

3.2 Možnosti tvorby webu

Možností, jak si vytvořit vlastní webovou stránku je dnes spousta. I zde platí staré pořekadlo, „za málo peněz málo muziky“. Čili pokud tvorbě nechcete věnovat čas, nebo peníze, nemůžete mít profesionální prezentaci se všemi požadovanými vlastnostmi.

Nejjednodušší variantou je využití hotové služby pro vytvoření webu. Funguje to tak, že vyplníte formulář, ve kterém si zadáte název své subdomény, vyplníte základní údaje o webu a své kontaktní údaje a web si jedním kliknutím vytvoříte. V lepším případě si vyberete z několika šablon, které můžete třeba barevně přizpůsobit. Rozložení prvků na stránce je obvykle dané, stejně jako použitá písma atp. Výhoda je v tom, že stránky máte „odkliknuté“ doslova za pár minut a stačí vám pouze naplnit je obsahem. Jste ale limitování šablonou webu, nemožností spouštět vlastní scripty atp.

Druhá, dnes již poměrně málo používaná varianta je použití nějakého šablonovacího systému. Vytvoříte si tedy opravdu vlastní webovou stránku, kterou můžete libovolně měnit a upravovat, navzdory tomu, že neznáte žádný šablonovací jazyk. Tento způsob by se dal přirovnat k programování pomocí funkčních bloků. Pouze víte co od kterého bloku očekáváte, ale neznáte žádné příkazy, žádný programovací jazyk. V tomto případě si tedy graficky upravujete web a zároveň v druhém okně vidíte, jak se vám mění HTML kód. Ten samozřejmě můžete také upravovat a výsledek se vám naopak zobrazí v grafickém okně. Nevýhoda je v tom, že program nerozpozná úplně všechny vaše záměry a můžou se zde vyskytovat některé redundantní tagy, nebo některé části nemusí být vyřešeny čistě, dle pravidel správného kódu, resp. syntaxe. Jedná se o offline program, který není nijak závislý na Internetu. Typickým zástupcem zmíněného softwaru je Microsoft Frontpage, dnes již aktualizován jako Microsoft Expression Web.

Další, aktuálně velmi žádanou možností, je použití tzv. „redakčního systému“, označovaného také zkratkou CMS (content management systém). Jedná se o hotový, zpravidla volně šířitelný systém, který je možné nejen využívat, ale dále modifikovat s použitím vlastních scriptů, nebo z dostupných balíčků (pluginů), které tvoří autor systému, nebo komunita, která kolem systému vznikla. Více o redakčních systémech (RS) je popsáno v kapitole 3.7 redakční systémy.

Vzhledem k tomu, že majoritní část našeho předmětu tvoří programování, se budeme zabývat poslední částí a tou je naprogramování si vlastního webu. Jedná se o časově, i

znalostně nejnáročnější možnost, nicméně u této varianty si můžete přizpůsobit úplně vše, nejste ničím omezeni.

3.3 Software pro tvorbu webu

Čistě teoreticky lze napsat celý web, frontend i backend v jakémkoliv textovém editoru, např. v poznámkovém bloku. Scripty se spouští na serveru a uživatel tak nepotřebuje žádné vývojové prostředí obsahující překladač, který slouží pro převod jeho kódu (vyššího programovacího jazyka) do strojového kódu. FTP si můžeme otevřít přes webové prostředí našeho poskytovatele webhostingu. Čili základní vybavení by mohlo obsahovat webový prohlížeč + libovolný textový editor.

V praxi se však snažíme práci zefektivnit a dostupné nástroje používáme pro vyšší programátorský komfort a rychlejší práci. Dostupných programů je celá řada. Níže jsou uvedené některé z nich, které jsou vyzkoušené a pro práci vyhovující. Ideální je mít jeden program, ve kterém kód napíšete a zároveň jedním klikem odešlete na server.

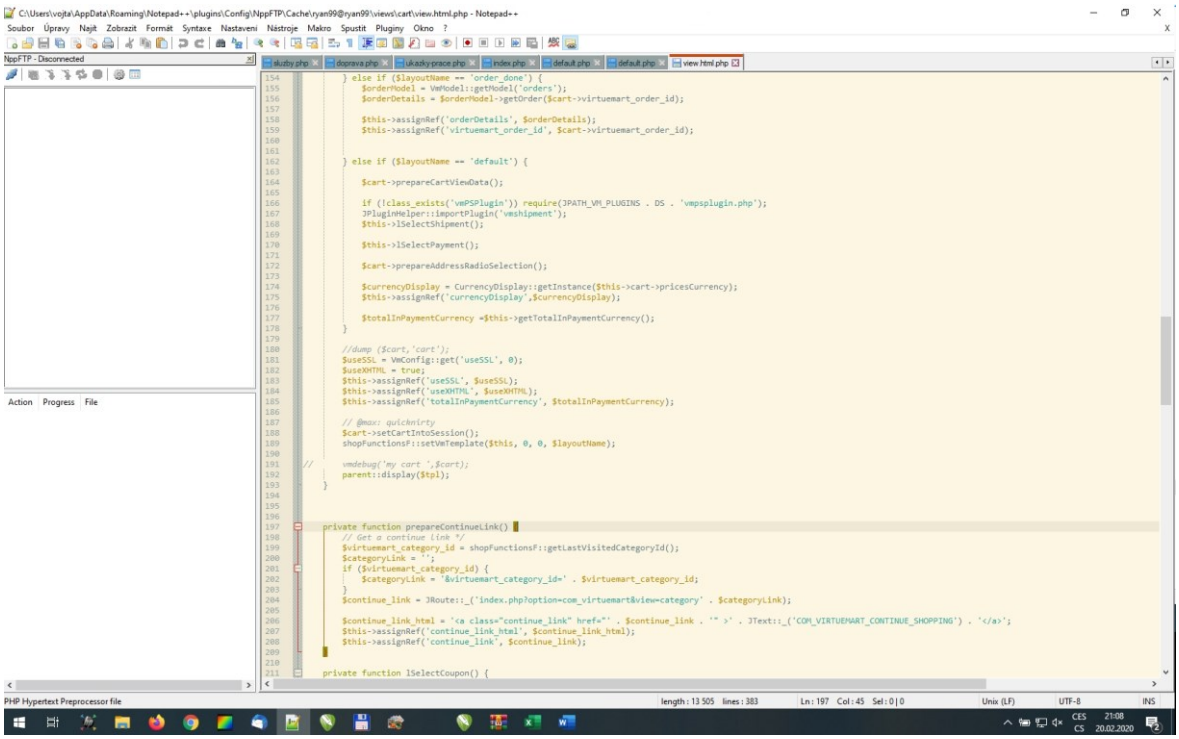
Program, který slouží pro zápis programovacího jazyka označujeme jako IDE (Integrated Development Environment) a značí nám vývojové prostředí pro konkrétní jazyk. Zásadní rozdíl oproti textovému editoru (kterým je např. poznámkový blok, word a další) je v tom, že je jazyku přizpůsobený. Má k dispozici seznam proměnných, operátorů, podmínek, funkcí, procedur atd. a na základě něj je dokáže:

- Barevně odlišit (lepší přehlednost kódu)
- Napovídat (začneme-li psát „ar“, napoví nám např. array)
- Čísluje řádky
- Označovat bloky kódu (např uzavřené do složených závorek)

3.3.1 Notepad++

Program vhodný jak pro šablony (HTML + CSS), tak pro programování v PHP. Dokáže barevně odlišovat různé části kódu, napovídat a po instalaci pluginu můžete scirpty odesílat rovnou na server. Stejně tak můžete na Vašem serveru po automatickém připojení vytvářet

soubory a složky, nahrávat soubory, vkládat nové scripty atd. Univerzální IDE, které je zdarma.



Obrázek 2 – Notepad++

3.3.2 PSpad

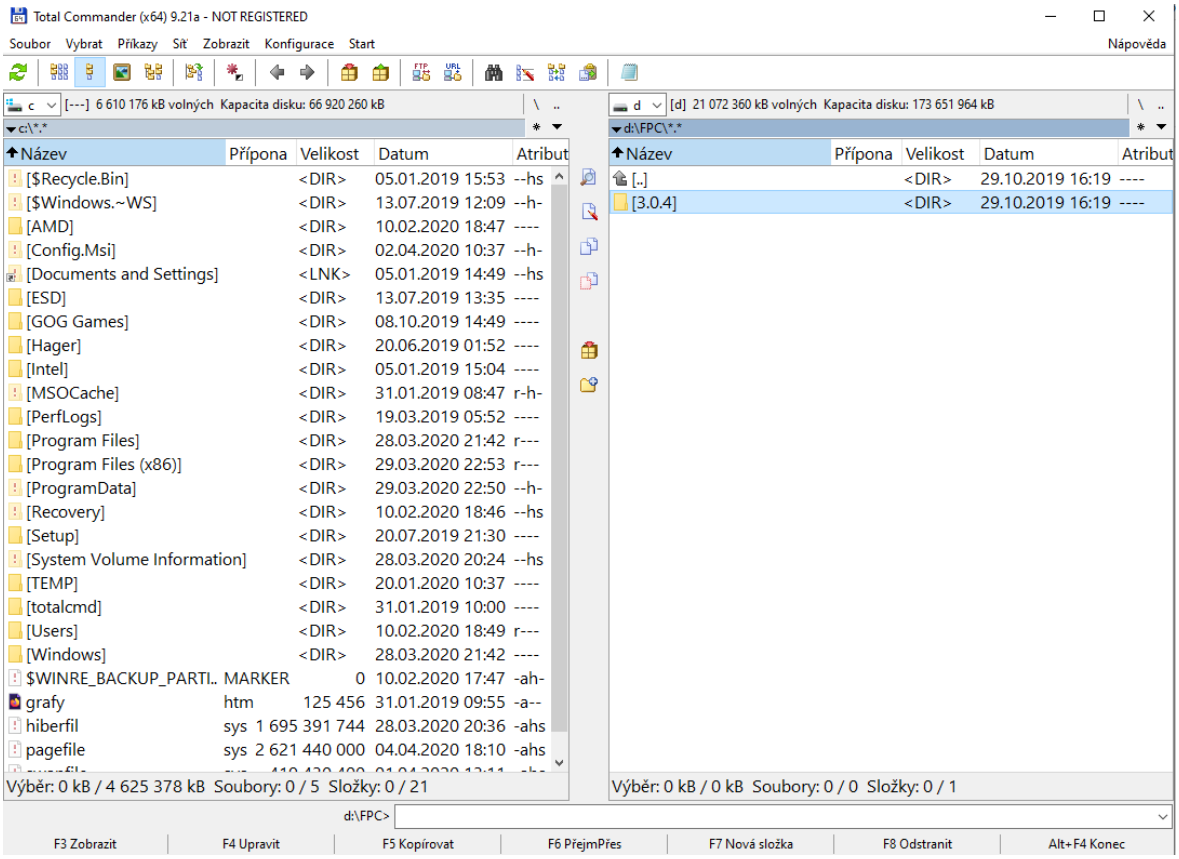
PSPad editor *je volně šiřitelný (freeware) univerzální editor pro MS Windows. To znamená, že si na něj nemusíte brát půjčku, nemusíte platit žádné peníze, protože je zadarmo. Neplatíte-li poplatky, ušetříte na dovolenou, kterou pak můžete pohodlně strávit někde v Karibiku.*
PSPad využijí všichni, kteří:

- *pracují s prostým textem - velké možnosti formátování textu*
- *vytvářejí webové stránky - obsahuje řadu unikátních funkcí, které ušetří spoustu času*
- *programují a potřebují IDE pro svůj kompilátor - odchytávání a parsování výstupu kompilace, integrace helpu, porovnávání verzí...*

[7]

3.3.3 Total Commander

Total commander je souborový manažer dostupný ve 30 denní shareware verzi, poté je nutná registrace. Mimo přehledné práci se soubory a složkami obsahuje už v základní verzi FTP modul, pomocí nějž je možné se připojit na server a pracovat se soubory a složkami také tam. Soubory můžete prostřednictvím total commanderu i otevřít v libovolném editoru (implicní je poznámkový blok), např. ve zmíněném Notepadu++, nebo PSpadu.



Obrázek 3 – Total Commander

3.4 Šablonovací jazyky

3.4.1 HTML

HTML (Hypertext Markup Language) je klíčovým jazykem pro tvorbu webových stránek. Nejedná se o programovací, nýbrž značkovací, šablonovací jazyk. Jednotlivými značkami (tagy) definujeme strukturu dokumentu, podobně, třeba jako v textovém editoru máme nadpisy, formátování textu (kurzíva, tučně), odstavce, atd. Úplně stejně funguje i webový dokument. Mimo výše popsaných elementů v HTML deklarujeme externí soubory, vkládáme obrázky, vytváříme formuláře, navigaci a strukturovaný obsah celé stránky. Platí zde určitá pravidla, tagy máme párové (např. nadpis je ohraničen na začátku, i na konci), nebo nepárové (např. obrázek). Internet, webové stránky, i samotný jazyk HTML se neustále vyvíjí, nicméně základ a princip jazyka zůstává neměnný. S HTML tedy definujeme rozložení webu a jeho obsahovou část. Pomocí CSS definované tagy „nastylujeme“.

Existuje možnost, jak se HTML vyhnout. Je možné se s ní setkat při přispívání na Internetové fórum, nebo kdekoliv na Internetu, kde se vkládá formátovaný text. Jedná se o tzv. WYSIWYG editory (what you see, it what you get), které nám HTML převedou do grafické podoby. Stejně jako v textovém editoru typu Word pouze stiskneme příslušné tlačítko a text bude tučně, jako nadpis atp. Tento editor na webu ale obsahuje pouze základní funkce. Je možné použít i speciální software popsaný v kapitole 3.3., který nám grafickou podobu na HTML tagy převede. Nutno dodat, že není vždy přesný a kdokoliv to s webovými technologiemi myslí vážně, měl by se alespoň se základy HTML jazyka seznámit.

3.4.2 HTML 5

Na HTML 5 čekali vývojáři již několik let. Změny byly opravdu potřebné a webové stránky se pomocí nové technologie stávají opět lepšími, přehlednějšími z hlediska kódu a pro vývojáře i jednoduššími. Mezi vítané změny patří zjednodušení doctype. Už není třeba řešit přepínání prohlížeče do qurick módu, ani kopírovat dlouhé Transitional a Strict doctype. Zavedena byla jednoduchá značka (tag) „<!DOCTYPE html>“. Změn dostáli i webové formuláře, kde lze blíže specifikovat obsah jednotlivých polí. Mimo dalších drobnějších změn, odstranění nepoužívaných tagů vývojáři také řešili problém tzv. předivování. Nebyl to problém pouze velkých modifikovaných systémů, kdy kodér v podstatě styluje hotovou HTML kostru, ale několik bloků zanořených v sobě obsahoval téměř každý druhý

web. V podstatě neexistovala jiná možnost jak oddělit základní části obsahu webu, jako je obsah, záhlaví, zápatí a menu.

Specifikace HTML5 poskytuje léčbu ve formě nových sémantických značek popisujících obsah, jež označují. Protože tolik vývojářů ve svých designech vytvořilo postranní panely, záhlaví, zápatí a části stránky, specifikace HTML5 uvádí nové značky specificky navržených k rozdělení stránky do logických celků. Vrhněme se tedy do práce s těmito elementy. Společně s HTML5 můžeme pomoci konečně vymýtit předivování. Mimo těchto strukturálních značek si povíme o elementu meter a probereme, jak můžeme využít možnosti nových vlastních atributů v HTML5, takže se můžeme věnovat vkládání obsahu do elementů místo toho, abychom naháněli třídy nebo existující atributy. Shrnuto v kostce, zjistíme, jak použít ty pravé značky vykonávající tu správnou práci.

[8]

Nové elementy:

- header (hlavička stránky, neplést s head)
- footer (zápatí stránky)
- nav (oblast s navigací stránky)
- section (logická sekce stránky)
- article (článek)
- aside (doplňující obsah)
- meter (míra v rámci rozsahu, ukazatel např. 25% ze 100%)

3.4.3 CSS

Cascading Style Sheets, neboli česky kaskádové styly je nedílnou součástí webové šablony (frontend). Pomocí HTML tagů si definujeme obsahovou část, vč. „kostry“ webu a pomocí CSS jednotlivým tagům přidělíme určité vlastnosti. Pomocí CSS můžeme měnit barvu (textu, nebo třeba pozadí části stránky), měníme pozice a rozložení jednotlivých elementů (např. hlavičku umístíme 20px od horní hrany atp.), definujeme typ písma, velikost, nebo měníme zarovnání textu. Prvkům definujeme velikost na stránce, měnit styly můžeme také formulářům, v CSS 3 můžeme objekty i otáčet.

Mimo čisté CSS můžeme využít i frameworků, což jsou knihovny vytvořené pro urychlení ručního stylování a některé můžou i zlepšit výkon celého webu. Kodérovi, který tvoří šablonu webu usnadňují také responzivitu webu. Nevýhodou je nutnost používat a učit se „něco dalšího“. Pro profesionální kodéry se ale tato časová investice rozhodně vyplatí. Mezi nejpoužívanější frameworky patří Bootstrap a Pure.

3.5 Programovací jazyky

3.5.1 PHP

V samých začátcích internetu jak ho známe byli webové stránky statické. Měli za úkol prezentovat určité informace, internet sloužil jako takový online „katalog“. Jelikož je svět kolem nás dynamický a technologie to umožňovala, weby získávali více a více interaktivních prvků, až dospěly do podoby velkých automatických systémů, internetových obchodů a administračních systémů jak je známe dnes.

Věk statických webových stránek je pryč. Po mnoho let byla síť WWW oblastí v níž několik jednoduchých, vzájemně propojených stránek HTML utvářelo webový server. Dnešní uživatelé však očekávají stránky, jež jsou průběžně aktualizovány a nabízejí vlastní zážitky. Kromě toho správci webových serverů požadují, aby byla aktualizace a údržba stránek snazší. Z těchto a mnoha dalších důvodů už je tvorba webového serveru pomocí statických souborů HTML nepřijatelná. Síť WWW je nyní místem pro dynamické, často databázemi řízené webové aplikace.

[9]

Vznik jazyka PHP se datuje do roku 1994, kdy byli první aplikace psané ještě v jazyce C. Tento jazyk prošel dlouhým vývojem, dnes je k dispozici ve verzi 7.3. Jedná se o programovací jazyk, nicméně výstupem nejsou spustitelné programy, nýbrž scripty, které zpracovává interpreter PHP na aplikačním serveru a uživateli vrací pouze HTML výstup. Proto je v literatuře často jazyk označován jako scriptovací. Z hlediska abstrakce se jedná o vyšší programovací jazyk, který je vhodně přizpůsoben svému použití a spoustu věcí programátorovi ulehčuje. Obsahuje dynamický typový systém, čili ani datovými typy se programátor v PHP nemusí příliš zabývat.

```php
*/
function show_blog_form( $blogname = '', $blog_title = '', $errors = '' ) {
    if ( ! is_wp_error( $errors ) ) {
        $errors = new WP_Error();
    }

    $current_network = get_network();
    // Blog name.
    if ( ! is_subdomain_install() ) {
        echo '<label for="blogname">' . __( 'Site Name:' ) . '</label>';
    } else {
        echo '<label for="blogname">' . __( 'Site Domain:' ) . '</label>';
    }

    $errmsg = $errors->get_error_message( 'blogname' );
    if ( $errmsg ) {
        ?>
        <p class="error"><?php echo $errmsg; ?></p>
        <?php
    }

    if ( ! is_subdomain_install() ) {
        echo '<span class="prefix_address">' . $current_network->domain . $current_network->path . '</sp
    } else {
        $site_domain = preg_replace( '|^www\.|', '', $current_network->domain );
        echo '<input name="blogname" type="text" id="blogname" value="' . esc_attr( $blogname ) . '" max
    }

    if ( ! is_user_logged_in() ) {
        if ( ! is_subdomain_install() ) {
            $site = $current_network->domain . $current_network->path . __( 'sitename' );
        } else {
            $site = __( 'domain' ) . '.' . $site_domain . $current_network->path;
        }

        printf(
            '<p>(<strong>%s</strong>) %s</p>',
            /* translators: %s: Site address. */
            sprintf( __( 'Your address will be %s.' ), $site ),
            __( 'Must be at least 4 characters, letters and numbers only. It cannot be changed, so choos
        );
    }
```

Obrázek 4 – Ukázka PHP kódu

Podle neoficiálních údajů některých měřících služeb používá ke svému fungování PHP 65% webových aplikací. Dalších 20% tvoří webů tvoří statické webové stránky a zbývajících 15% ostatních jazyky (zejména ASP.NET). Vzhledem k faktu, že je jazyk interpretován a je možné je před měřením skrýt, nemusí být data přesná, nicméně je pravděpodobné, že nebudou daleko od reality.

3.5.2 ASP.NET

ASP.NET je webový framework, stručně řečeno se jedná o sadu knihoven, které umožňují tvorbu webových aplikací v jazyce C#. Knihovny obsahují hotová řešení mnoha základních problémů, které ve webových technologiích vyvstávají. To jsou např. bezpečnost, autentifikaci uživatele, práci s databází, správu formulářů a podobně. ASP tedy není programovací jazyk, programovat budeme v C# a budete tedy potřebovat základní znalosti tohoto jazyka

[10]

ASP.NET je často využívané zejména na Windows serverech a využívá databáze MS SQL. Mezi tvůrci webových stránek ale není tolik rozšířený a oblíbený jako PHP.

3.5.3 Automatická tvorba dokumentace

Protože dokumentace tvoří důležitou část procesu efektivního vytváření kódu a jeho správy, bylo vynaloženo hod-ně úsilí, aby se našlo dobré řešení, které by opravdu pomáhalo vývojářům tento proces automatizovat. A skutečně, v současné době jsou už k dispozici pokročilá dokumentační řešení pro všechny mainstreamové programovací jazyky, včetně PHP. Nástroj s názvem phpDocumentor (www.phpdoc.org) je open-source projekt, který usnadňuje proces dokumentace tím, že převádí komentáře vložené do zdrojového kódu do různých snadno čitelných formátů, mezi něž patří HTML a PDF. Nástroj phpDocumentor pracuje tak, že analyzuje zdrojový kód aplikace a hledá v něm speciální komentáře zná-mé jako DocBlocks. DocBlocks se používají ke zdokumentování veškerého kódu uvnitř aplikace, včetně skriptů, tříd, funkcí, proměnných a dalších prvků, obsahují lidem srozumitelná vysvětlení spolu s formalizovanými deskrip-tory, jako jsou jméno autora, verze kódu, prohlášení o copyrightu, návratové hodnoty funkcí a mnoho dalších věcí. I když jste úplně začínající programátor, vřele se doporučuje, abyste se obeznámili s pokročilými dokumentačními řešeními a zvyknuli si je standardně používat i v jednoduchých aplikacích.

[11]

3.6 Databázové systémy

Nedílnou součástí běžného dynamického webu je databáze. Využijeme ji všude tam, kde chceme ukládat nějaký obsah a dále s ním pracovat. Nemusí se nutně jednat o databázi zákazníků v internetovém obchodu, ale můžeme naši DB (databázi) využít pro uložení textů pro jednotlivé stránky webu. Celý web se poté načítá dynamicky a nemusíme mít několik html souborů, jak např.:

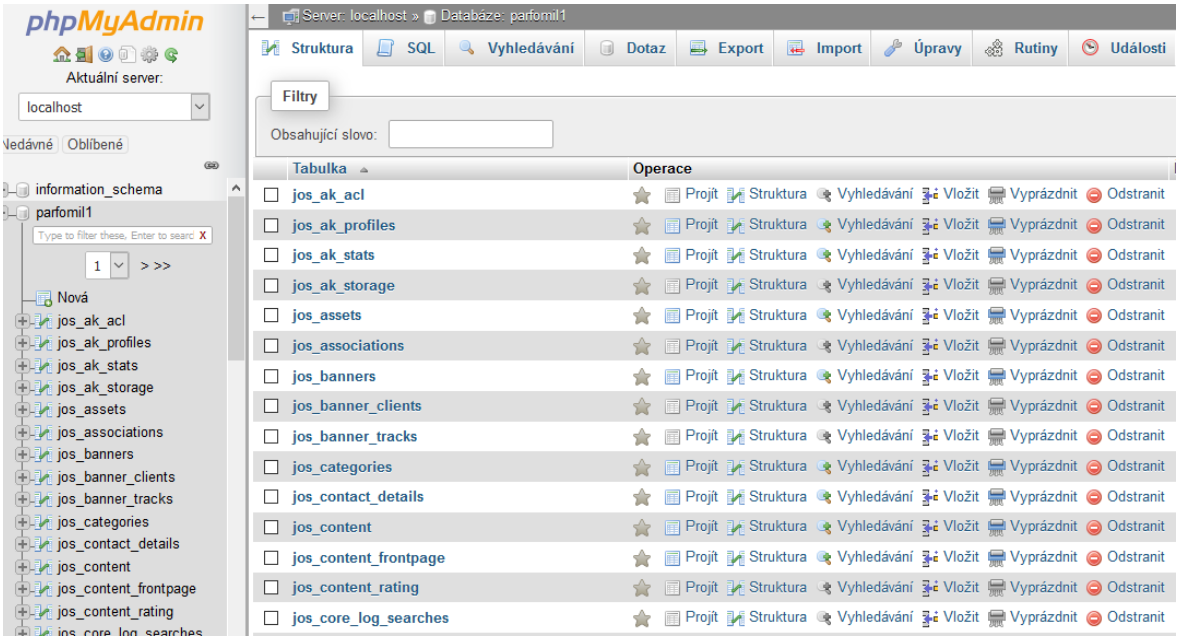
- Index.html
- O_nas.html
- Kontakt.html
- Sluzby.html

Stačí nám pouze jedna úvodní stránka (index.php) a zbytek podstránek obsloužíme právě pomocí databáze. V databázi si založíme tabulku např. články a v ní bude:

ID	Název	Text	Datum	URL
1	O nás	Nějaký text…	23.2.2020	o_nas
2	Kontakt	Telefon 732…	25.2.2020	kontakt
3	Služby	Nějaký text..	25.2.2020	sluzby

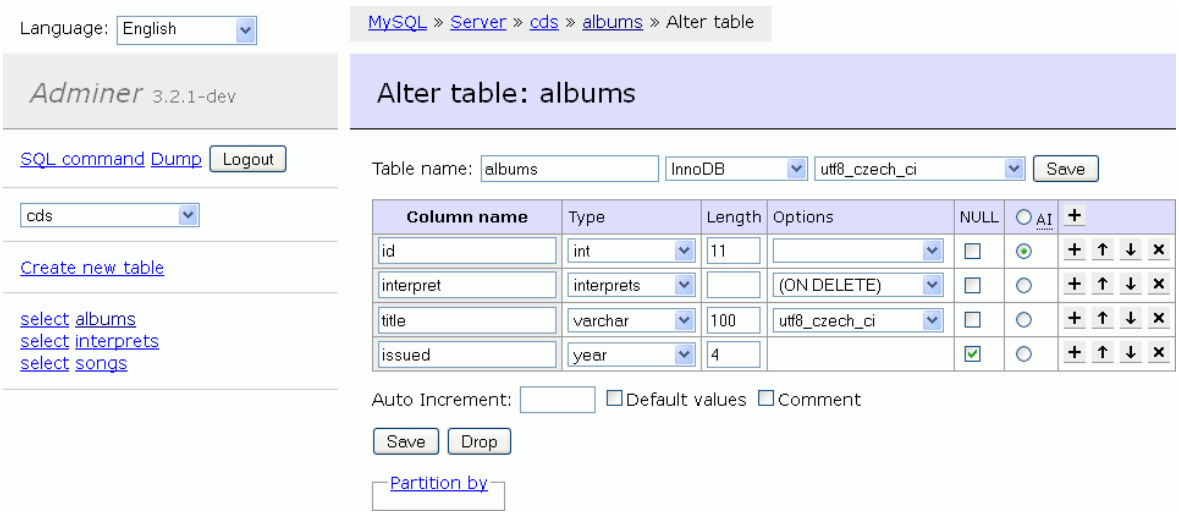
Tabulka 1 – Demonstrace databázové tabulky

Struktura je velmi zjednodušená, ve skutečnosti by mohla obsahovat údaje jako titulek webu (který nemusí být shodný s názvem), název pro menu (který může být např. kratší než celý název), description (popisek pro meta tag stejného názvu), jméno autora atd. Nicméně klíčové položky naše tabulka obsahuje a to je ID, nebo-li identifikátor, jedinečné označení záznamu, tak aby nedošlo k záměně záznamu za jiný. Vždy tak můžeme specifikovat, že chceme vybrat záznam s ID 4 a ten se v tabulce bude vyskytovat pouze jeden. Názvy, i ostatní položky tabulky totiž mohou být shodné. K tomuto účelu se využívá primární klíč, který je nastaven jako unikátní. Samozřejmě v tomto jednoduchém příkladu by primární klíč mohl být i název, nebo URL, ale v praxi se používá pro všechny tabulky číselný údaj, identifikátor. Dále obsahuje název článku, samotný text, datum a URL.



Obrázek 5 – Nástroj pro správu databáze phpMyAdmin

Společně s PHP scripty se v oblasti webových technologií ve většině případů používá databáze mySQL, kde jako grafické prostředí pro přístup do databáze používáme, a na většině hostingových služeb je dostupný phpMyAdmin. V případě, že trváte na co nejmenším zatížení serveru, je možné použít nástroj nazvaný Adminer českého autora Jakuba Vrány.



Obrázek 6 – Nástroj pro správu databáze Adminer

Autor jako výhody uvádí přehlednější uživatelské rozhraní, lepší podporu vlastností MySQL, vyšší výkonost a bezpečnost. Mezi důležitými vlastnostmi svého nástroje prezentuje zejména:

- ***Připojení** k databázovému serveru pod zadaným uživatelským jménem a heslem*
- *Výběr **databáze**, vytvoření nové databáze*
- *Seznam sloupců, indexů, cizích klíčů a triggerů tabulky*
- *Změna názvu, úložiště, porovnávání, auto_increment a komentáře **tabulky***
- *Změna názvu, typu, porovnávání, komentáře a výchozí hodnoty **sloupců***
- *Přidání a smazání tabulek a sloupců*
- *Vytvoření, změna, smazání a vyhledávání podle **indexů** včetně fulltextových*
- *Vytvoření, změna, smazání a propojení seznamů podle **cizích klíčů***
- *Vytvoření, změna, smazání a získání dat z **pohledů***
- *Vytvoření, změna, smazání a zavolání **uložených procedur a funkcí***
- *Vytvoření, změna a smazání **triggerů***
- ***Výpis dat** s možností vyhledávání, třídění a omezení počtu vypisovaných záznamů*
- *Vložení, úprava a smazání **záznamu***
- *Podpora všech **datových typů**, práce s `BLOB` přes nahrávání souborů*
- *Provedení libovolného **SQL příkazu** zadaného přímo nebo nahraného ze souboru*
- ***Export** struktury tabulek, dat, pohledů, uložených procedur a databází do SQL nebo CSV*
- ***Schéma struktury databáze** s vazbami podle cizích klíčů*
- *Seznam **procesů** s možností jejich ukončení*
- *Přehled **uživatelů a práv** s možností jejich nastavení*
- *Přehled **proměnných** s odkazy do dokumentace*
- *Správa **událostí** a **rozdělených tabulek** (MySQL 5.1)*
- *Schéma, sekvence, uživatelské typy (PostgreSQL)*
- *Široké možnosti přizpůsobení*

[11]

3.6.1 Datové typy

Hodnoty se samozřejmě liší, u každé předpokládáme trochu jiný obsah. Stejně tak jako u programování v některých jazycích, které nemají dynamický typový systém jako PHP, je nutné určit pro každou proměnnou, co bude obsahovat (číslo, znak, pole znaků, logickou proměnnou), stejné je to i v případě databází. U malého webu se nic zásadního nestane, pokud nezvolíme místo textu číslo a opačně, ale u větších projektů jde především o úsporu místa v úložišti. Pokud si pro název vyhradíme pole znaků o délce 50 000, budeme zbytečně zabírat místo o velikosti 49950 znaků, protože název obvykle nebude delší než např. 50 znaků. Datové typy by jsme tedy mohli rozdělit na:

- Číselné datové typy
- Textové datové typy
- Datum a čas
- Logický datový typ (boolean)

Číselné datové typy

Název	Popis	Alokovaná paměť	Rozsah čísel	Příklad
Integer	Celá čísla	4 B	- 2147483648 až 2147483647 Případně 0 až 4294967295	INT(X)
Float	Čísla s desetinnou čárkou	4 B	±1,175494351E−38 až ±3,402823466E+38	FLOAT(X,Y) X = délka, Y počet míst za desetinou čárkou
Double	Velká čísla s desetinou čárkou	8 B	±2,2250738585072014E-308 až −±1,17976931348623157E+308	DOUBLE(X,Y)

Tabulka 2 – Databáze: číselné datové typy

Textové datové typy

Ukládání textů v MySQL řešíme obvykle pomocí tří základních

datových typů. Těmi jsou CHAR (pole znaků), VARCHAR (pole znaků o proměnné délce) a TEXT používaný pro delší bloky textu. V drtivé většině běžných aplikací si vystačíte pouze s těmito třemi datovými typy. Srovnání v následující tabulce:

Název	Maximální velikost	Alokovaná paměť	Příklad
CHAR (X)	255 B	X B (X bajtů)	CHAR (25)
VARCHAR (X)	255 B	X + 1B	VARCHAR(200)
TEXT	65535 B	X + 2B	TEXT

Tabulka 3 – Databáze: textové datové typy

Pro znaky s ASCII tabulky platí 1 znak = 1 B. Používáme-li české znaky (např. UTF kódování), může jeden znak zabírat více bytů. Zásadní rozdíl mezi CHAR a VARCHAR je v tom, že pokud budeme mít 25 znakový CHAR a vložíme pouze 3 znaky, např. „ABC“, v paměti stejně bude zabírat 25 B. Naproti tomu stejný VARCHAR (25) při uložení 3 znaků „ABC“ bude alokovat pouze 3 + 1, tedy 4 B. Poslední zmíněný textový datový typ TEXT se používá pro delší texty, např. pro obsah našich výše zmíněných článků. Alokované místo v paměti je u tohoto typu automaticky upraveno.

Datum a čas

Název	Alokovaná paměť	Formát	Příklad
Date	3 B	CCYY-MM-DD	Date(1922-02-05)
Time	3 B	hh:mm:ss	Time(15:22:00)
Datetime	8 B	CCYY-MM-DD hh:mm:ss	Datetome(1922-02-05 15:22:00)

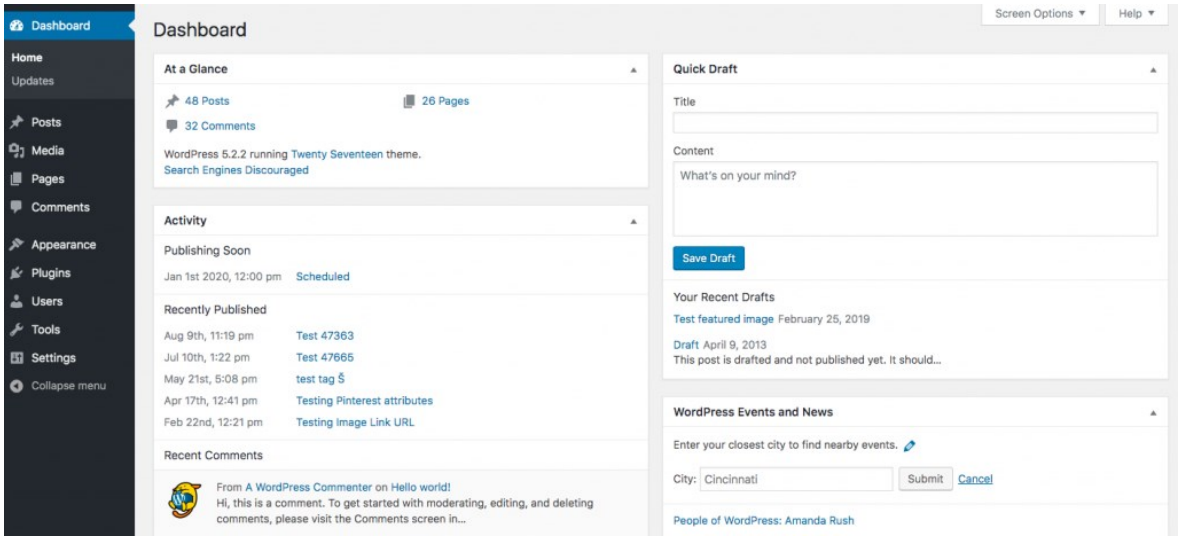
Tabulka 4 – Databáze: datové typy pro datum a čas

3.7 Redakční systémy

Zajímavou alternativou pro znalé uživatele je použití RS (redakčního systému), který je vytvořen zkušenými programátory a v případě volně šiřitelného RS (opensource) také otestován tisíci uživateli po celém světě. Protože existují desítky opensource RS, není důvod se zabývat jinými, komerčními RS, které najdou své využtí ve velkých projektech, kde už z pohledu klienta není rozdíl ve využití interního RS určité firmy, nebo naprogramování vlastního systému stejnou firmou. Tyto systémy jsou samostatnou aplikací, kterou si může uživatel (programátor) libovolně upravovat (v případe opensource systému), přidávat pluginy třetích stran (např. napojení na EET, objednávkový systém atp.), případně upravovat jádro systému k obrazu svému – tím samozřejmě může dojít ke kolizi v případě novější verze systému v případě aktualizace. Systémy tohoto typu bývají často také označovány jako CMS (content managet systém – systém pro správu obsahu). Tato práce se opensource redakčními systémy zabývá pouze okrajově, proto jsou uvedeny pouze ty nejpoužívanější v prostředí českého internetu.

3.7.1 Wordpress

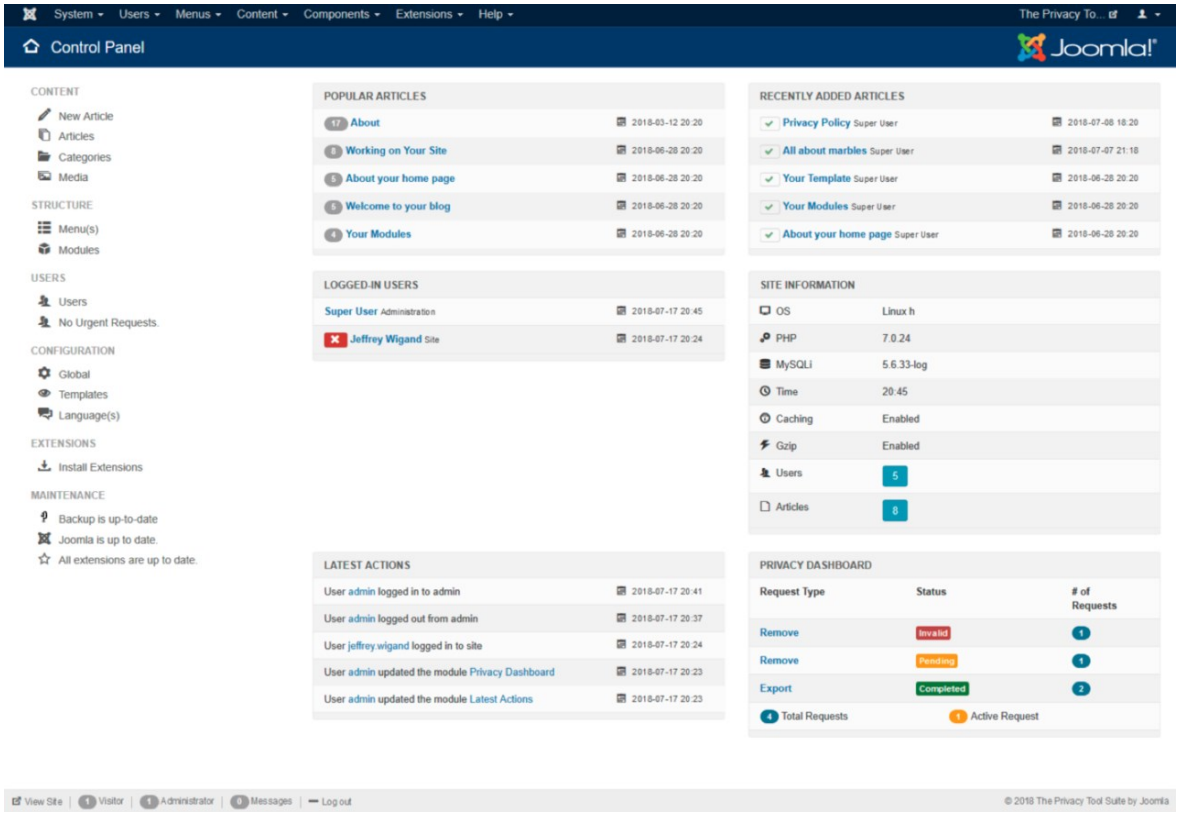
Tento nejprve blogovací systém vznikl v roce 2003 a v současné době je jedním z nejuniverzálnějších CMS v oblasti webu. Pro technicky méně zdatné uživatele má intiutivní a dobře zpracovanou administraci, vč. instalace systému. Pro systém jsou k dispozici tisíce šablon (themes) a na internetu existuje (i v češtině) spousta návodů na instalaci (systému, pluginů), i na samotnou obsluhu. Díky pluginům je možné z původně blogovacího systému vytvořit elektronický obchod, diskusní fórum, komunitní web a další. Pro bezproblémový běh je nutné sledovat aktualizace a instalovat bezpečnostní pluginy. Systém je založený na PHP a MySQL.



Obrázek 7 – Administrační rozhraní CMS systému Wordpress

3.7.2 Joomla

Joomla vznikla o dva roky později, v roce 2005. Jedná se o složitější systém než wordpress, je tedy vhodnější spíše pro pokročilé uživatele. Stejně jako wordpress pracuje na objektovém PHP a využívá databází MySQL. Opět i pro joomlu existuje spousta doplňků, vč. známého pluginu „virtuemart“, který využívají tisíce internetových obchodů. Joomla lze použít, nicméně není úplně vhodná pro malé jednoduché weby. Mimo jiné pro tento systém není k dispozici tolik šablon (themes, templates) jako pro výše zmíněný konkurenční wordpress. Joomla zvládá také autentizaci uživatelů pomocí OpenID.



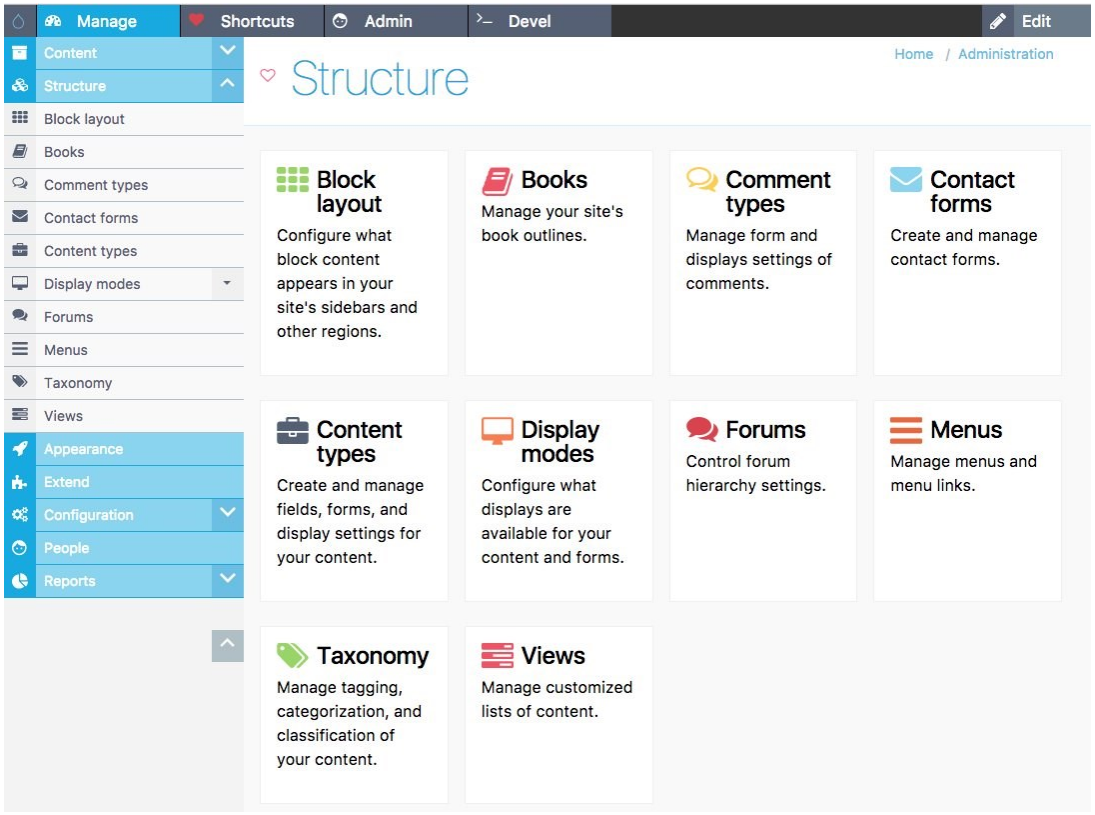
Obrázek 8 – Administrační rozhraní CMS systému Joomla

3.7.3 Drupal

Tento systém vznikl již v roce 2001 a je vhodný pro zkušené vývojáře, kteří jsou schopni aplikovat i složitější požadavky svých klientů. Systém, stejně jako dva předchozí, uvedené výše pracuje na technologii PHP s databází MySQL.

Drupal je považován za nejflexibilnější a nejvýkonnější CMS, který je vhodný především pro vytváření komplexních a technicky propracovaných podnikových webů.

[12]



Obrázek 9 – Administrační rozhraní CMS systému Drupal

3.8 Domény a DNS

Doména je adresa webu, kterou zadáváme do adresního řádku webového prohlížeče. Internet při svém vzniku pochopitelně žádné domény neznal a ke své funkci propojení pomocí počítačové sítě tyto adresy vůbec nepotřebuje. Byly vytvořeny kvůli zapamatovatelnosti pro uživatele. Díky doménám tak není nutné uvádět IP adresu webu (tedy cílového serveru), ale jakoukoliv posloupnost znaků abecedy a čísel (použití diakritiky je kvůli duplicitě vyloučeno). Domén máme několik řádů, při čtení začínáme vždy „od konce“. Máme-li doménu http://blog.grapro.cz , pak je doménou prvního řádu národní doména „CZ“. Doména druhého řádu, kterou lze vlastnit jako fyzická, nebo právnická osoba je „grapro“ a tzv. subdoména, neboli doména 3. řádu je „blog“. Celkově tato URL odkáže uživatele na doménu třetího řádu. Majitel domény 2. řádu si subdomény vytváří sám a „neomezeně“ v rámci serveru.

DNS (domain name systém – systém doménových jmen) server tvoří důležitou funkci a v podstatě zajišťuje překlad URL adresy do podoby IP adresy. U sdílených hostingů je na jedné IP adrese více webů a server musí rozlišit, který uživatel požaduje.

4 BEZPEČNOST HESLA V PROSTŘEDÍ WEBU

4.1 Jak zvolit heslo

Drtivá většina běžných uživatelů volí taková hesla, která si lehce zapamatují. Jedná se o podněty z jejich okolí, jména rodinných příslušníků, data narození, RČ, jméno oblíbeného sportovce atp. To je samozřejmě špatně, neboť taková hesla lze snadno uhodnout pomocí tzv. slovníkové metody. V případě hesla platí, že nejlepší heslo je takové, které nikdo nezná, nejedná se o běžně používané slovo, resp. o běžně používanou kombinaci znaků. Dnes na nás hodně tlačí světový jazyk angličtina. Je pochopitelné, že anglicky mluví podstatně více lidí než češtinou a proto při tvorbě hesla nemusíte být in a vymýšlet anglické heslo, ale naopak, mnohem bezpečnější z celosvětového pohledu bude použití češtiny. V světových slovnících pro útočníky se bude mnohem častěji vyskytovat „apple“, než „jablko“.

4.2 Autentizace

Jakákoliv aplikace, kde potřebujeme pracovat s konkrétními uživateli potřebujeme nějakým způsobem ověřit identitu uživatele. V reálném světe identitu ověřujeme pomocí dokladu totožnosti, např. občanského průkazu. Na webu a v mobilních aplikacích obvykle používáme jméno, email atp. + heslo. Heslo tvoří kombinace znaků z dostupné abecedy a v ideálním případě by jej měl znát pouze majitel(é) daného účtu. Pro proniknutí do aplikace se poměrně často používá právě tato část aplikace a jednotlivé útoky si rozebereme v následující kapitole.

4.3 Typy útoků

4.3.1 Hrubá síla

Budeme předpokládat běžně používané heslo skládající se z 26 znaků anglické abecedy (bez háčků a čárek). Takové heslo je stále dobrým příkladem, protože některé systémy neumožňují tyto znaky zahrnout do Vašeho hesla. Anglická abeceda má 26 znaků. Vzhledem k tomu, že nám záleží na pořadí (jiné heslo bude „les“ a jiné „sel“), použijeme variace (jsou vždy uspořádané) Písmena se nám můžou opakovat (např. l**o**k**o**m**o**tiva obsahuje 3x stejný znak), budou to tedy variace s opakováním.

$V'(k,n) = n^k$

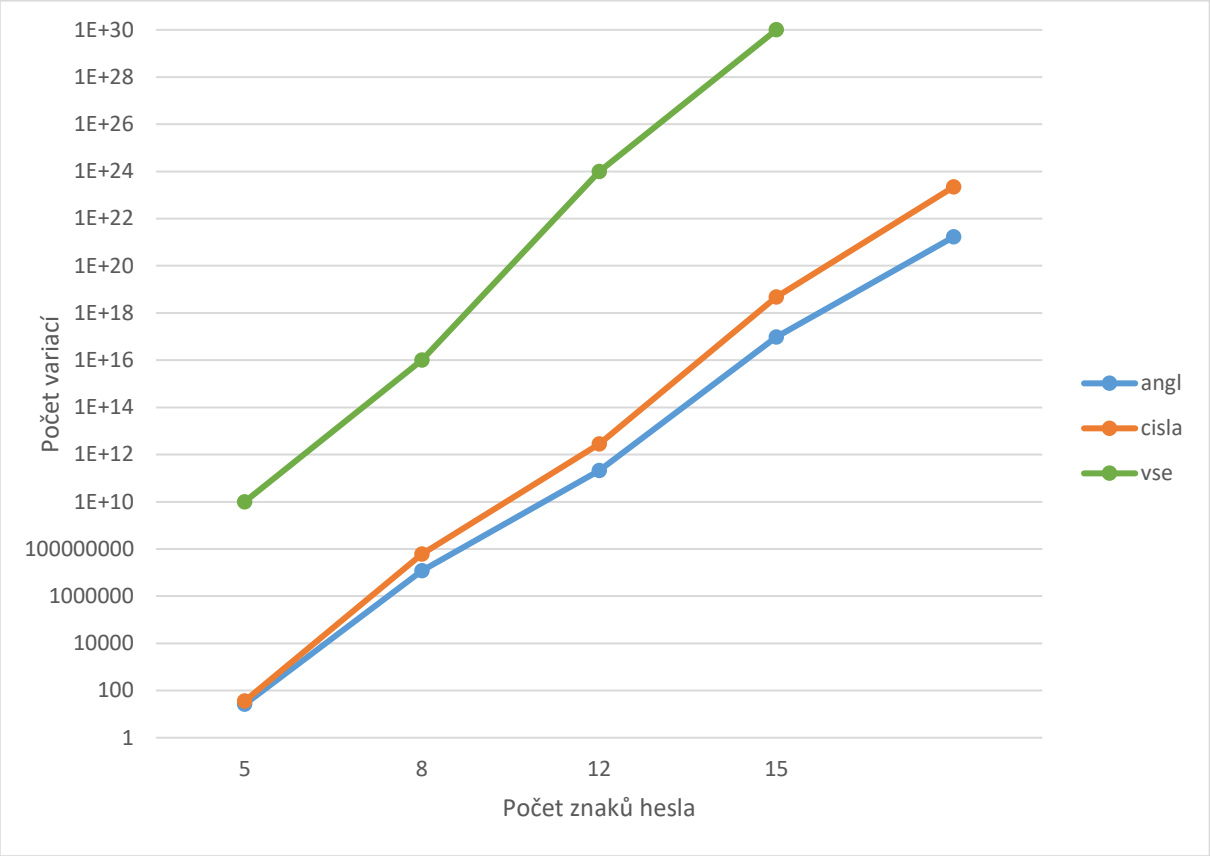
Kde „n“ je počet písmen, ze kterých budeme vybírat (pro nás 26), a „k“ počet písmen, které obsahuje naše heslo, tedy 6. Vyjde nám přibližně 308 milionů variací. To je poměrně vysoké číslo, pokud si budeme zkoušet hesla psát na papír. V případě výkonných počítačů, které běžně dokáží porovnat i 50 milionů řetězců za sekundu by tak prolomení trvalo asi 6 vteřin. To už tak skvěle nezní.

Vytvořme si malou tabulku, ve které vidíme jak dlouho nám trvá prolomit jaké heslo.

Délka hesla	Pouze znaky anglické abecedy (26)	Znaky + čísla (36)	Znaky české abecedy, velká písmena + speciální znaky (100)
5	12 milionů	60 milionů	1000 milionů
8	208827 milionů	$28*10^{11}$	$10*10^{15}$
12	$95*10^{15}$	$47 * 10^{17}$	$10*10^{23}$
15	$16*10^{20}$	$22 * 10^{22}$	$10*10^{29}$

Tabulka 5 – Počet porovnání řetězců za vteřinu

Data vyneseme do grafu:



Obrázek 10 - Závislost možných variací na počtu znaků hesla

Samozřejmě útočník používá velmi sofistikované algoritmy, které jsou optimalizované na základě četnosti výskytu určitých znaků, podle čehož je algoritmus upřednostní a pravděpodobnost dřívějšího nalezení hesla je tak vyšší. Stejně tak útočník předpokládá že velké písmeno bude na začátku a čísla na konci. Útočník může, na základě dobré znalosti daného jazyka ve kterém je heslo pravděpodobně napsáno, algoritmus velmi dobře optimalizovat a značně tak urychlit odhalení hesla. Např. v češtině lze předpokládat, že je málo pravděpodobné, že v jednom slově budou dvě po sobě jdoucí samohlásky, stejně tak může omezit méně častá písmena, resp. nechat jejich testování až na závěr (např. pro jeden případ z 10) atp.

Uživatelé se proti výše zmíněné optimalizaci útoku hrubou silou mohou bránit jednoduše tak, že do hesla použijeme pseudonáhodné znaky, velká písmena zvolíme třeba dvě, a ne na předvídatelné pozici, čísla nebudou ordinální (tedy uspořádaná 123) atp. Můžeme k tomu

využít generátor „náhodných“ hesel, problém potom ale může být se zapamatováním takového hesla.

4.3.2 Slovníková metoda

U tohoto útoku útočníci využívají již připravené databáze nejpoužívanějších hesel a cílí na běžného uživatele u kterého předpokládají, že by mohl jako heslo zvolit nějaký obecně známý pojem. Technicky je tato metoda stejná, jako útok hrubou silou, jen nezkoušíme obměňovat jednotlivé znaky, ale rovnou celá slova. Neexistuje zde jistota, že heslo oběti ve slovníku určitě bude (u útoku hrubou silou je zde 100% jistota), ale je pravděpodobné, že určité procento uživatelů používá typická hesla a díky slovníkové metodě je útočník dokáže prolomit mnohem dříve, než u útoku hrubou silou.

Důležitý je také výběr správného slovníku. Ten musíme přizpůsobit na základě získaných informací o oběti – zejména z jaké pochází země, jaké má zájmy atp. Slovníky můžou být jednoduché, ale také velmi sofistikované, kdy za běžná slova přidávají také letopočty související s danou tématikou, obsahují velké první písmeno atp.

Pochopitelně nejefektivnější je pro útočníka použít nejdříve rychlou slovníkovou metodu a teprve až selže, přejít na útok hrubou silou.

4.3.3 Rainbow tables

Tento typ útoku nelze použít přímo na autentizaci konkrétní aplikace, ale používá se při úniku dat z databáze. Útočník např. zjistí heslo k databázi, nebo se dostane na FTP serveru, případně získá fyzicky harddisk, kde jsou data uložena atp. Problém je v tom, že hesla nejsou (neměla by být) v databázi uložená jako tzv. plain text, tedy standardní čitelný text typu „moje heslo“, ale měla by být uložena jako „nečitelný“ hash. Hashovací funkce je jednosměrná, tedy z hotového hashe nelze zpětně spočítat jeho původní text, což je pro útočníka problém. Aby útočník nemusel zkoušet generovat všechna hashe, použije právě rainbow tables.

Jedná se o databáze, řekněme tabulky, které obsahují předem spočítané hashe různých hesel. Máme-li tedy k dispozici databázi hashovaných hesel, resp. hash nějakého hesla, velmi snadno dokážeme díky rainbow tables zjistit jeho původní znění, tzv. plain text.

4.3.4 Sociální inženýrství

Tento pojem je poměrně těžké uchopit zcela exaktně. Existuje velké množství definic. Některé považují za sociální inženýrství jakékoliv promyšlenější napadení násilnou penetrací, druhá vlna naopak za sociální inženýrství považuje jakýkoliv podvod v oblasti IT bezpečnosti.

Sociální inženýrství je nejjednodušší a zároveň nejúčinnější metoda, jak získat podrobné informace o cíli. Soc. inženýrství je metoda, kdy se útočník zaměří na svou oběť s cílem získat důvěrné a cenné informace. Získané informace následně zneužije ve svůj prospěch

[13]

Útočníci využívají toho, že v dnešním světe jsme zvyklí komunikovat elektronicky na denní bázi a setkáváme se tak s různými požadavky, dotazy atp. Útočník se snaží zapadnout do tohoto „chaosu“ a vylákat z oběti citlivé informace, aniž by tušila, že se jedná o sofistikovaný a připravený útok. Mnohdy tedy únik informací není vůbec prozrazen a ani při zpětném prošetřování si oběť neuvědomí svou chybu.

Soc. inženýrství velice často cílí na emoce lidí, zneužívají je a převrací ve svůj vlastní prospěch. Jiní naopak navazují mezilidské vztahy s cílem vytvořit si přátelské pouto, které vyvolá v oběti důvěru.

[14]

Útočník musí mít v tomto případě nejen technické, ale také psychologické a sociální znalosti. Využívá psychologické manipulace k prosazení svých požadavků. V souvislosti se sociálním inženýrstvím je nutné definovat ještě pojem sociotechnika, která je mylně považována za synonymum k sociálnímu inženýrství. Ve skutečnosti se jedná o jeden z možných nástrojů sociálního inženýrství, obvykle nekybernetického charakteru, pomocí kontaktu v reálném světě, viz. definice:

Sociotechnika je ovlivňování a přesvědčovaní lidí s cílem oklamat je tak, aby uvěřili, že sociotechnik je osoba s totožností, kterou předstírá a kterou si vytvořil pro potřeby manipulace. Díky tomu je sociotechnik schopný využít lidi, se kterými hovoří, případně dodatečné technologické prostředky, aby získal hledané informace.“

[15]

4.4 Opatření z technického hlediska

4.4.1 Vynucená síla hesla

Asi nejjednodušší metoda, kterou můžeme implementovat je vynucení síly hesla. Při registraci, nebo změně hesla uživatelem, zadaný řetězec kontrolujeme a zjišťujeme zda obsahuje minimální počet znaků - např. 8, zda obsahuje velké písmeno a číslici.

Pokud chceme jít ještě dále, můžeme také kontrolovat, zda se nejedná o příliš časté heslo pomocí vlastní databáze často používaných hesel (typicky heslo, password, různá jména – Adelka16 atp.). Můžeme také ověřovat, zda není velké pouze první písmeno (jak útočník předpokládá, viz. výše), zda není číslice na konci a další předpokládané potencionální chyby.

Implementace na webu, například v PHP není složitá, během několika málo hodin ji dokáže implementovat každý průměrný programátor.

4.4.2 Uzamykání účtů

Další možností, tentokrát jak příliš neobtěžovat uživatele, ale zabránit potenciálním útokům, je dočasné uzamknutí uživatelského účtu. Změnu provedeme pouze v přihlašování, nikoliv v registraci. Za normálního stavu uživatel změnu ani nepozná. U každého uživatele hlídáme (např. si vytvoříme sloupec v databázi), kolikrát se zkoušel přihlásit a po jaké době. Nastavíme např. maximálně 3 pokusy během jedné minuty.

Vezmeme-li v úvahu, že pří útoku hrubou silou, tzv. brutal force attacku dokáže útočník vyzkoušet desítky milionů hesel za vteřinu, jedná se o obrovskou bariéru a komplikaci pro útočníka. V tomto případě musí čekat, než bude moci zadávat další 3 hesla a prolomení hesla se tak stává v podstatě nemožným. Graf složitosti prolomení hesla by v tomto případě byl lineární a stoupal každých 60 vteřin o 3. Pokud se jedná o citlivou aplikaci, je možné každé takové krátkodobé zablokování nahlásit administrátorovi a samozřejmě zaslat útočníkovu IP adresu, i když je pravdou, že ta nám obvykle nepomůže, útočník obvykle:

- Použije veřejnou Wifi, kterou sice dohledáte, ale připojují se tam desítky počítačů denně
- Použije proxy server – v podstatě se připojí přes cizí IP adresu, která jeho požadavky zrcadlí

- VPN služba (virtuální soukromá síť, při připojení k internetu jako zdroj připojení je vidět některý z VPN routerů poskytovatele, nikoliv ten uživatelský)
- Síť TOR

Některé banky jdou tak daleko, že při 3 neúspěšných přihlášeních účet kompletně zablokují a je možné se znovu přihlásit pouze po rozhovoru na kontaktní zákaznické lince, kde je nutné ověřit totožnost volajícího, např. řadou osobních otázek zadaných při registraci.

4.4.3 Logy, IP adresy

Z vizuální kontroly logů můžeme vyčíst potencionální útok. Toto opatření je ale dosti pracné a hodí se spíše v případech, kde nemáme možnost jiného zabezpečení.

Co se týče IP adres, můžeme jednak vysledovat útočníka, což je ale dosti problematické, viz. výše. Útočník by musel být poměrně hodně neinformovaný, aby se tímto způsobem prozradil. Je zde také možnost přihlašovat se např. přes konkrétní IP adresy, kdy je Vaše IP adresa zadána, nebo načtena při registraci a z jiné sítě se jednoduše nepřihlásíte.

Nevýhod to má hned několik. Jednak se nepřipojíte třeba z domu, pokud máte zadanou firemní IP adresu, účet nemůžete s nikým sdílet a v neposlední řadě můžete mít dynamickou IP adresu, čili její přesné znění obdržíte od DHCP serveru při připojení do sítě, má určitou dobu expirace atd. Čili rozhodně nebude stejná po celou dobu užívání dané aplikace.

I tyto problémy se dají ošetřit. Když jsem byl požádán o naprogramování webu, do kterého měli přístup klienti pouze se zaplaceným účtem, musel jsem nějak vyřešit fakt, že by spousta zákazníků účty přeprodávalo a na jeden placený účet se přihlašovalo X dalších klientů zdarma. Vyřešili jsme to omezením na IP adresu s tím, že přihlásit se bylo možné i s jinou IP adresou, nicméně byl zapsán údaj do sessions a během další půl hodiny se nemohl přihlásit nikdo jiný. Mělo to tu nevýhodu, že se pravý majitel nemohl zároveň připojit z práce a za 10 minut z domu, nicméně jednalo se o natolik specifickou aktualizaci, že to zásadně nevadilo a značně se omezilo zneužívání účtů.

4.4.4 Dvoufaktorová autentizace

Při standardním přihlašování používáme obvykle dva údaje – jméno a heslo. Jméno může být také email, příjmení, atp. Nicméně, nejedná se o žádný citlivý a chráněný údaj. Pokud jej píšete před ostatními na projektoru, je viditelné, stejně tak v databázi je uloženo jako čistý text, kdokoliv si jej může přečíst, jedná se o veřejný údaj. K autenzitaci (tedy ověření, že jste to opravdu Vy) dochází pomocí hesla. Pokud dojde k prolomení hesla, útočník získává přístup k Vašemu účtu. Dvoufaktorová autentizace nespoléhá pouze na jedno heslo, ale ověřuje dva faktory, čímž se stává teoreticky z hlediska autentizace samotné dvojnásobně bezpečnou (podmínkou ovšem je správně ošetřený program). Ověřovat uživatele můžeme v podstatě na základě tří oblastí:

- Znalosti - kódu, hesla, pinu
- Biometrických údajů – otisk prstu, rozeznání hlasu, sítnice oka atp.
- Externího předmětu – mobilní telefon, klíč na flash disku, fyzický klíč, token atd.

Jak bylo uvedeno výše, první kategorii používáme v drtivé většině internetových aplikací, stejně tak pin u svého telefonu, nebo kreditní karty. Dvoufaktorová autentizace spočívá v implementaci ještě jedné kategorie, v případě internetových a mobilních aplikací obvykle fyzického tokenu, nebo mobilního telefonu. Při přihlášení tedy zadáte veřejný údaj – jméno a dva neveřejné údaje z různých kategorií, typicky heslo a kód z fyzického tokenu, nebo kód zaslaný SMS zprávou na Váš mobilní telefon. Jsou-li zastoupeny tyto dvě kategorie, útočníkovi tak nestačí prolomit pouze heslo, ale musí mít k dispozici i token, nebo mobilní telefon oběti.

Je pochopitelné že dvoufaktorová autentizace je náročnější na implementaci a také vyžaduje další aktivitu uživatele, což při několikanásobném denním přihlašování může být nepohodlné. Obvykle se tedy používá u citlivých aplikací, kde má ochrana dat přednost před uživatelským komfortem. Typicky se jedná o bankovní systémy, nebo dálkové řízení nějakého zařízení.

Dvoufaktorovou autentizaci lze také implementovat přímo do aplikace telefonu, takže není nutné složitě přecházet do SMS zpráv a opisovat ručně kód. Komfort uživatele je tak zachován a bezpečnost systému zesílena.

4.5 Hashovací funkce

Hashovací funkce je používána ke skrytí původního textu (např. hesla) do takové podoby, která je nečitelná. Samotný algoritmus není utajován, jedná se o jednosměrnou funkci. Na rozdíl od šifrování, hash nelze nijak dešifrovat. Při ukládání hesla jej nepotřebujeme dešifrovat, stačí nám ze zadaného řetězce vytvořit hash a ten poté ověřit s původním hashem. Inverzní funkce ani nesmí existovat, jinak by bylo pro útočníka jednoduché vzít hash a zjistit původní heslo.

4.5.1 MD5

Hashovací funkce řady MD (Merkle-Damgardova konstruce) fungují už od roku 1989 a konkrétně MD5 je používána od roku 1992 a některými systémy je používána dodnes, ačkoliv se její použití už dlouhou dobu nedoporučuje.Český kryptolog Vlastimil Klíma v roce 2016 publikoval algoritmus, který je schopen najít kolize MD5 na běžném počítači během jedné minuty. Tuto metodu nazval tunelování. Výstupem funkce MD5 je hash o délce 128 bitů, tedy 32 znaků. Dnes už se tato funkce vzhledem k reálné odhalitelnosti (rainbow tables) a kolizním algoritmům (dva odlišné texty vygenerují stejný otisk) víceméně nepoužívá.

4.5.2 SHA

SHA, neboli Secure Hash Algorithm je skupina hashovacích funkcí navržených Národní bezpečnostní agenturou v USA a obsahuje celkem 5 druhů hashů: SHA1, SHA-224, SHA-256, SHA-384 a 512.

4.5.3 Kryptoanalýza SHA-1

Pro ideální hashovací funkci platí, že nelze najít zprávu, která by byla vzorem k danému hashi. Porušení tohoto pravidla, může být docíleno hrubou silou, vyhledáním 2L výpočtů, kde L je počet bitů hashe. Tento útok se nazývá vzorový (preimage attack). Druhým kritériem je, že nelze najít stejný hash pro dvě rozdílné zprávy. Na toto pravidlo lze zaútočit pomocí narozeninového útoku (birthday attack), který vyžaduje pouze 2L/2 výpočtů. Tento útok je založen na narozeninovém paradoxu, který říká, že pravděpodobnost nalezení páru dvou lidí, kteří mají stejné datum narození, je u skupiny 23 lidí, kteří jsou náhodně vybráni, 50%. U skupiny 57 lidí je to již 99% a dále pravděpodobnost roste až ke 100%. Nalezení takového páru se nazývá kolize. Síla hashovacích funkcí je obvykle porovnávána se symetrickými

šiframi jako polovina délky hashe. Tedy síla SHA-1 byla původně odhadována na 80 bitů. Poté, co bylo oznámeno úplné prolomení algoritmu SHA-0 12. června 2004, se objevily pochybnosti odborníků o zavádění SHA-1 do nových kryptografických systémů. Po CRYPTO 2004 (mezinárodní kryptografická konference), kde bylo prolomení SHA-0 zveřejněno, NIST oznámilo, že plánují nahradit SHA-1 algoritmem SHA-2 a jeho variantami do roku 2010. V únoru roku 2005 byl zveřejněn útok vedený Xiaoyun Wang, Yiqun Lisa Yin a Honbo Yu. Jejich útok nalezl kolizi v plné verzi SHA-1 a vyžadoval méně než 269 operací. Na setkání CRYPTO 2006, Christian Rechberger a Christophe De Canniere prohlásili, že objevili kolizní útok, který dovolí útočníkovi vybrat alespoň část zprávy. Největším znepokojením u těchto útoků je to, že připravují cestu pro mnohem účinnější útoky. Proto se považuje přechod k silnějším hashovací algoritmům jako rozumný. Některé použití hashovacích algoritmů, jako je uložení hesel, je minimálně ovlivněno kolizními útoky. Konstrukce hesel účtů vyžaduje vzorový útok a přístup k originálnímu hashi, což nemusí být jednoduché. V případě podpisu dokumentů musí útočník vytvořit dvojici dokumentů, jeden neškodný a jeden škodný, a dát neškodný dokument k podepsání držiteli soukromého klíče.

[16]

4.5.4 Password_hash (Bcrypt + Solení)

Aktuálně nejvhodnější variantou pro zabezpečení hesla na webu je algoritmus „bcrypt“, případně „argon2“ v kombinaci se solením. Aktuálně nejpoužívanější hasovací funkcí je bcrypt, používají jej přední webové stránky. Argon 2 je relativně čerstvý a pro uživatele bcryptu zatím nebyl důvod přechodu na Argon 2.

Vzhledem k tomu, že výkon dnešních počítačů roste a je možné zřetězené zpracování na více procesorech najednou, bcrypt jednoduše využívá zpomalení výpočtu. Běžný uživatel rozdíl nepozná, je mu víceméně jedno, jestli výpočet proběhne za 5, nebo 50ms a pro útočníka je to výrazné zpomalení. Vývojáři jsou si vědomi neustálého pokroku a lidské vynalézavosti, takže navrhli funkci „password_hash“, která do výstupního hashe vloží také verzi použitého hashovacího algoritmu, zpomalení a sůl. Nebude potom do budoucna problém přejít na jinou hashovací funkci.

Ukázka vytvoření hashe na základě defaultního hashovacího algoritmu (např. Bcrypt, nebo Argon2):

$hash = password_hash("zadaneHeslo123", PASSWORD_DEFAULT);

Nebo v případě, že chceme použít konkrétně Bcrypt:

$hash = password_hash(zadaneHeslo123, PASSWORD_BCRYPT);

Poté ověření správnosti zadaného hesla:

if(password_verify ("zadaneHeslo123" , $hash));

Výsledný hash potom vypadá např. takto:

$6b$20$i51asd519df12d5sc15ojgpv/o9JBs1SA9v4aD3s5df4IkjdA8qVa95

Jako oddělovač slouží znak dolaru „$“ a následně lomítko. První dva znaky za dolarem značí verzi algoritmu, další dva znaky za oddělovačem tzv. „cenu“, která nám určuje jak moc bude algoritmus zpomalen, následuje sůl a za lomítkem červeně hash.

4.5.5 Sůl nad zlato

Jak vidíme, i historické, či pohádkové výroky pronikají do světa internetu, konkrétně do zabezpečení hesel.

Při hashování hesla se na vstup navíc ještě přidává nějaký další řetězec, takzvaná sůl. Sůl je pokaždé jiná, výsledný hash je tak i ***pro stejná hesla pokaždé jiný****.*

Solí může být obecně cokoliv. Není nutné, aby byla tajná. Naopak, obvykle jde o veřejnou hodnotu. Jediný rozumný požadavek je, aby se lišila navzájem mezi jednotlivými uživateli.

Častým řešením bývá ***náhodně generovaná sůl****. Tu je pak potřeba v databázi ukládat do speciálního sloupce vedle uživatelského jména a výsledného otisku. Funkce* crypt() *v PHP si (v závislosti na použitém algoritmu) dokonce n-znakovou sůl uchovává přímo v rámci výsledného otisku jako jeho prvních n znaků.*

Nevýhody náhodné soli jsou spíše praktického charakteru, například je pak složitější realizovat Challenge/Response autentizaci, protože se sůl musí pokaždé posílat uživateli společně s výzvou.

Výrazně praktičtější mi proto osobně přijde ***jako sůl použít samotné uživatelské jméno****. Nemusí se pro ni definovat samostatný databázový sloupec. Nemusí se nikam posílat, protože uživatelské jméno je vždy rovnou k dispozici.*

[17]

Bohužel v případě útoku hrubou silou nám ani solení nepomůže. Solení hesel tedy určitě nemůžeme považovat za jedinou metodu zabezpečení, je nutné ji kombinovat s dalšími technikami.

5 KURIKULÁRNÍ DOKUMENTY

V současné době střední školy pracují se školními vzdělávací mi programy (dále jen ŠVP), které si každá škola zpracovává sama na základě příslušeného RVP (rámcový vzdělávací program). Ve svém ŠVP zohledňuje individuální vzdělávací podmínky dané školy a může se tak u různých středních škol stejného zaměření lišit. ŠVP schvaluje ředitel školy a zodpovídá za soulad s RVP. ŠVP je k dispozici k nahlédnutí každému rodiči i pedagogovi.

Každý ŠVP vychází z RVP, který určuje jakýsi rámec, ze kterého ŠVP následně vychází. RVP garantuje povinný rámec učiva, který je nutno pro daný obor naplnit. RVP vydává MŠMT (Ministerstvo školství, mládeže a tělovýchovy) a pro každý obor vzdělávání vydává samostatný dokument.

Vzhledem k tomu, že moje práce je obecná, nebudu se zaměřovat na ŠVP konkrétní školy, ale budu vycházet z obecných požadavků RVP pro obor Informační technologie 18 – 20 – M/01. Dílčí části mé práce mohou být využity i na gymnáziích, průmyslových, nebo obchodních školách, případně dalších typech SŠ.

5.1 Rámcový vzdělávací program (RVP)

RVP, které vydává MŠMT je možné získat na webu NÚV (Národní ústav pro vzdělávání), kde je možno nahlédnout do jednotlivých RVP podle úrovně vzdělávání a podle konkrétních oborů. V RVP pro obor Informační technologie 18-20-M/01 je pro oblast webových stránek vymezena část v odborných kompetencích (3.2. e). Žáci si mají vytvořit kompetenci:

Programovat a vyvíjet uživatelská, databázová a webová řešení, tzn. aby absolventi:

- *algoritmizovali úlohy a tvořili aplikace v některém vývojovém prostředí;*
- *realizovali databázová řešení;*
- *tvořili webové stránky.*

Přesnější popis nalezneme v kurikulárních rámcích, což jsou:

Kurikulární rámce vymezují závazný obsah všeobecného a odborného vzdělávání a požadované výsledky vzdělávání. Obsah vzdělávání se člení na vzdělávací oblasti a obsahové okruhy (viz kapitoly 1.2 a 1.3). Kurikulární rámce rozpracuje škola ve školním vzdělávacím programu do vyučovacích předmětů, popř. dalších vzdělávacích aktivit a činností, a to s ohledem na požadavky nebo možnosti trhu práce i studijní předpoklady a zájem žáků. Podle charakteru oboru vzdělání lze odborné vzdělávání

rozpracovat také směrem k určité oblasti odborných činností. ŠVP může být zaměřen např. na vývoj aplikací a programování, realizaci a správu webových řešení, počítačových sítí, operačních systémů, aplikací, HW a další. Výsledky vzdělávání jsou stanoveny jednotně pro všechny žáky, je však zřejmé, že kvalita (úroveň) jejich osvojení bude záviset také na učebních předpokladech a motivaci každého žáka. Výsledky vzdělávání vyjadřující žádoucí postoje a návyky žáků (afektivní cílové dovednosti), kterými je škola sice povinna žáka vybavit, ale nemůže zaručit jejich uplatňování v praxi, jsou vyjádřeny zpravidla v charakteristice jednotlivých oblastí a obsahových okruhůjako vzdělávací cíle, k nimž musí výuka směřovat. Požadavky stanovené pro oblasti všeobecného vzdělávání, kromě vzdělávání ekonomického, navazují na RVP základního vzdělávání.

[18]

Já se zaměřím na kurikulární rámec programování a vývoj aplikací, kde jsou zahrnuty webové technologie.

5.2 RVP pro oblast WWW stránek

PROGRAMOVÁNÍ A VÝVOJ APLIKACÍ

Cílem obsahového okruhu je naučit žáka vytvářet algoritmy a pomocí programovacího jazyka zapsat zdrojový kód programu. Žák porozumí vlastnostem algoritmů a základním pojmům objektově orientovaného programování, dále se naučí používat zápis algoritmu, datové typy, řídící struktury programu, jednoduché objekty a základní příkazy jazyka SQL. Podstatnou část vzdělávání v programování a vývoji aplikací představuje samostatná tvorba jednoduchých aplikací, statických a dynamických WWW stránek.

Výsledky vzdělávání	Učivo
Žák: -zná vlastnosti algoritmu; -zanalyzuje úlohu a algoritmizuje ji; -zapíše algoritmus vhodným způsobem;	1) Algoritmizace -význam, prvky algoritmu
-použije základní datové typy; -použije řídící struktury programu; -vytvoří jednoduché strukturované programy	2) Strukturované programování -datové typy -řídicí struktury
- rozumí pojmům třída, objekt a zná jejich základní vlastnosti; - použije jednoduché objekty;	3) Úvod do objektového programování - třída, objekt, vlastnosti tříd
-zná výhody použití jazyka SQL; -použije základní příkazy jazyka SQL;	4) Základy jazyka SQL -základní příkazy (SELECT, UPDATE, INSERT, DELETE)
-aplikuje zásady tvorby WWW stránek; -orientuje se ve struktuře HTML stránky; -vytvoří webové stránky včetně optimalizace a validace; -použije formuláře a skriptovací jazyk.	5) Tvorba statických a dynamických webových stránek

Tabulka 6 – RVP pro oblast webových technologií

[18]

II. PRAKTICKÁ ČÁST

6 PROJEKTOVÁ VÝUKA WEBOVÝCH TECHNOLOGIÍ NA SŠ

Praktická část této diplomové práce je zaměřena na výuku webových technologií na střední škole. **Tvorba webových stránek je natolik komplexní obor, že není reálné provádět výuku tohoto předmětu na střední škole pouze formou projektu, respektive projektových úkolů.** Hlavní osnovu výuky lze pořadím projektů stanovit, ale v kódování a programování je potřeba určitých znalostí a správných návyků k vychování dobrého programátora. Z výše uvedeného důvodu jsem se rozhodl podpořit projektovou výuku (která by jinak musela být natolik podrobná, že by téměř nesplňovala vlastnosti projektové výuky) výukou frontální, pomocí prezentací. Zda ji vyučující využije je z cela na jeho úvaze, nicméně jsem přesvědčen o tom, že má-li mít moje práce určitý přínos pro výuku na SŠ, je zcela na místě vytvořit komplexní výukový plán, který bude kombinovat forntální výuku a projektové pracovní listy pro stěžejní témata. Výstupem této diplomové práce je tedy komplexní soubor materiálů pro výuku WWW technologií zahrnující nejen samotnou projektovou výuku, ale také návrh frontální výuky vč. připravených prezentací a doplňujících úkolů. **Ústředním tématem práce bylo vytvoření pracovních listů pro výuku webových technologií – ty jsou připraveny jak k frontální, tak k projektové výuce.** Kurz je rozdělen do 16 výukových lekcí + 6 projektových úkolů ke stěžejním tématům. Ke každé lekci je připravena prezentace, podklady pro samostatnou práci, případně ukázky hotové samostatné práce. **Projektové úkoly, které jsou součástí navrhnuté výuky opakují, rozšiřují a doplňují téma probrané v prezentaci.** Lze je využít samostatně, bez prezentace, nebo naopak provést nejprve projektový úkol a následně si učivo shrnout pomocí prezentace. Součástí práce je i test, který společně se splněnými úkoly může sloužit k celkovému hodnocení. Je tak na každém vyučujícím, kterou část využije, případně zda ji upraví a v jakém rozsahu.

Cílem diplomové práce bylo také ověřit projektovou výuku tvorby WWW stránek v praxi. Výuku jsem realizoval na Obchodní akademii v Šumperku v rámci své učitelské praxe v 2. ročníku. Bohužel v souvislosti s pandemií způsobenou virem COVID-19 jsem měl možnost realizace pouze 2 hodin výuky, následně byly školy uzavřeny.

6.1 Obsah výuky

Výuka, má-li být efektivní musí probíhat v jistém konceptu, což odpovídá frontální výuce. Projektová výuka je velmi otevřená a ze své podstaty musí mít dostatečně široké téma. Studenti by se tak k některým, důležitým pasážím nemuseli dostat, přestože diskusí nad hotovými projekty je možné jejich úhel pohledu rozšířit. Například v algoritmizaci a programování je obvykle více cest jak vyřešit daný problém, nicméně, je důležité, aby se studenti nespokojili pouze s jedním řešením. Je důležité, aby se naučili správným zásadám a měli přístup k relevantním informacím, které jsou pro ně dobře vstřebatelné. U projektové výuky je zde prostor při závěrečné diskusi, při prezentaci jiného řešení spolužáka. Každopádně, v takovém případě není zaručeno a ověřeno, že student skutečně přemýšlí nad jiným řešením, nebo „usnul na vavřínech“ s vlastním, správně vyřešeným úkolem. Projektová výuka má velké přínosy pro rozvoj osobnosti, zodpovědnosti, komunikačních dovedností atd. Její zařazení je určitě vhodné. Kombinací frontální a projektové výuky se tato práce snaží dosáhnout co nejefektivnější výuky WWW technologií na SŠ.

Frontální výuka je rozdělena do dvou fází:

- Prezentace (přednáška)
- Zpracování samostatného úkolu

Projektová výuka je navrhnuta pro stěžejní témata, kterými jsou:

- Úvod do WWW technologií
- bezpečnost na internetu
- Tvorba frontendu
- algoritmizace
- Tvorba backendu
- Databáze

U prezentace považuji za velmi důležité, aby studenty zaujala, přitom působila profesionálně. Látka je studentům předkládána v přiměřeném tempu, úkoly jsou voleny dle tématu lekce, některé jsou navrhnuty pro řešení v rámci hodiny, některé jako samostatná práce (v hodině, případně dopracovat doma).

V rámci projektového vyučování se studenti učí samostatné práci, zodpovědnosti, podobně jako třeba v zaměstnání. Základní informace tak získají z prezentace vyučujícího a další část

potřebného učiva a znalostí získají během práce na projektu, kde si celou „kapitolu“ shrnou a vytvoří finální, vlastní produkt. Projektová výuka má za cíl upevnit získané znalosti z přednášky, na které volnou formou navazuje a doplnit jejich znalosti o další informace. Kombinaci projektové a klasické výuky považuji za velmi efektivní a vhodnou pro studenty SŠ.

Součástí projektové výuky je také spolupráce ve skupinách, kde si studenti vytýčí své sociální role a přijímají zodpovědnost za vlastní práci nejen před sebou samými, ale i před ostatními spolupracovníky ve skupině. Závěrem většiny projektových úkolů je také prezentace výsledků před ostatními, kdy mohou následovat dotazy a diskuse nad tématem, eventuelně hodnocení dané práce. Práci je možné provést různými způsoby, v případě dobrého způsobu je vhodné studenta pochválit, v případě funkčního, ale pomalého, zbytečně komplikovaného řešení je vhodné studenta upozornit a doporučit mu jiný způsob – případně k jinému způsobu dojít společně.

6.2 Cílová skupina

Výuka je zaměřena na studenty středních škol, zejména průmyslové školy, gymnázií s rozšířenou výukou informatiky, ale i ostatní typy středních škol, kde bude v rámci ŠVP zařazena tvorba WWW stránek, případně některé pasáže kurzu. Mimo výše zmíněných subjektů, kurz může využít každý, kdo má zájem naučit se základy webových technologií a tvorby webových stránek.

Kurz je strukturován tak, aby pojmul to nejzákladnější z frontend (kódování), i backend (programování) vývoje webu. Krátce se výuka zmiňuje také o databázích a bezpečnosti aplikací. Jako navazující by mohl být druhý kurz rozšiřující znalosti v jedné ze zmíněných oblastí, který by naplnil druhé pololetí výuky WWW technologií. V případě gymnázia, nebo odborné školy zaměřené na programování, kde se ale zaměřují např. na desktopové aplikace je kurz vhodným úvodem do problematiky. Studenti získají přehled v oblasti WWW technologií a pevný základ pro další, např. samostudium.

Projekty jsou navrhnuty tak, aby danou oblast shrnovali, eventuelně doplňovali znalosti, které studenti nestačili, nebo z nějakého důvodu nezvládli získat během klasické výuky.

6.3 Časová dotace

Výukový kurz je dle tempa žáků možné absolvovat během 19 týdnů a může plnohodnotně nahradit výuku informatiky na SŠ v jednom pololetí. Celkově je koncipován tak, aby jej bylo možné zvládnout za 38 vyučovacích hodin. Kurz počítá s tím, že některé úkoly studenti dopracují doma, čili je možné tempo přizpůsobit a více, či méně úkolů zadávat jako domácí práci.

Číslo lekce	Téma	Časová náročnost		
		Teorie	Cvičení	Celkem
Lekce č. 1	Úvod do WWW technologií	2	0	2
Lekce č. 2	Bezpečnost na internetu	1	0	1
Lekce č. 3	Tvorba šablony - obecně	1	1	2
Lekce č. 4	Tvorba šablony – HTML	1	1	2
Lekce č. 5	Tvorba šablony – CSS	1	1	2
Lekce č. 6	Vytvoření vlastního webu	0	4	4
Lekce č. 7	Shrnutí znalostí – test	1	0	1
Lekce č. 8	Základy algoritmizace	4	2	6
Lekce č. 9	Úvod do PHP	1	1	2
Lekce č. 10	První aplikace v PHP	1	1	2
Lekce č. 11	Větvení v PHP	1	1	2
Lekce č. 12	Cykly	1	1	2
Lekce č. 13	Práce s poli	1	2	3
Lekce č. 14	Vytvoření databáze	1	1	2
Lekce č. 15	Implementace databáze do aplikace	1	2	3
Lekce č. 16	Zabezpečení webu	1	1	2

Tabulka 7 – Seznam vypracovaných lekcí

Co se týče projektových úkolů, jejich celková délka je 23 vyučovacích hodin a je tak na vyučujícím, zda touto formou výuky nahradí standardní hodinu, zadá ji jako domácí úkol, nebo, bude-li mít dostatek prostoru využije obě formy výuky. Záleží na přidělené časové dotaci a rozsahu, v jakém se bude výuce WWW technologií věnovat.

Projektový úkol číslo	Téma	Časová náročnost
1	Úvod do tvorby WWW stránek	2x45min
2	Bezpečnost na internetu	3x45min
3	Tvorba šablony webu	4x45min
4	Poznáváme algoritmy	4x45min
5	Programujeme PHP formulář	6x45min
6	Spravujeme databázi	4x45min

Tabulka 8 – Seznam projektových úkolů

Využije-li vyučující všechny navrhnuté materiály (prezentace + projektové úkoly), jedná se celkem o 61 hodin, které lze absolvovat během jednoho školního roku, při časové dotaci 2h týdně.

6.4 Mezipředmětové vztahy

Kurz, i projektová výuka může být začleněna do informatických předmětů, případně některé jeho části do programování na odborné škole. Primárně se tedy jedná o projektovou výuku spadající do informatiky. V rámci mezipředmětových vztahů pracuje se znalostmi z českého jazyka (obsahovou náplň tvoří studenti sami), anglickým jazykem (programovací jazyky vychází z angličtiny) a matematikou (v příkladech se často objevují výpočty).

6.5 Rozvoj klíčových kompetencí

6.5.1 Kompetence k řešení problémů

Celá projektová výuka je postavena tak, aby řešila jednoduché úkoly (problémy) z oblasti WWW stránek. Studenti programují reálné aplikace, které jsou sice triviální, ale jsou plně funkční a použitelné v praxi. V případě nefunkčního programu (nebo šablony, která nevykazuje takové vlastnosti jaké má mít) analyzuje situaci a snaží se zjistit v čem je problém. Dokáže se poučit s chyb, jak vlastních, tak z chyb ostatních studentů, prezentovaných před třídou v rámci diskuse. Vyhodnotí, zda dokáže nastavený problém řešit samostatně, s pomocí informačních zdrojů, nebo s pomocí jiné osoby (spolužáka, vyučujícího).

6.5.2 Kompetence k učení

K získávání informací a učení vybírá vhodné metody a strategie, během řešení problémů řídí vlastní poznávání a učení. Nalezené informace třídí a analyzuje, experimentuje s nimi v rámci řešeného problému a kriticky posuzuje zvolená řešení. Dokáže rozeznat vlastní pokrok v práci a v učení, plánuje jednotlivé fáze řešení problému – získávání informací, analýza, zpracování projektu, experimentování, kontrola a testování.

6.5.3 Kompetence komunikativní

Při výuce algoritmizace a programování je velmi vhodnou výukovou metodou diskuse. Studenti se snaží přijít na možná řešení a zároveň se mohou poučit z chyb ostatních a vyvarovat se jim. Výuka algoritmizace a programování by měla být založena na diskusi. Stejně tak studenti mohou konzultovat své řešení s ostatními, případně s vyučujícím. A to nejen v závěru, po odevzdání, ale i v průběhu práce.

6.5.4 Kompetence sociální a personální

Student přispívá do diskuse v rámci řešeného problému, chápe možnosti spolupráce na řešení problému s ostatními. Vytváří si pozitivní představu o sobě samém na základě vyřešení, dílčího, nebo celého problému. Řídí svou činnost tak, aby dosáhl co nejlepšího výsledku a dosáhl tak pocitu sebeuspokojení a sebeúcty.

6.5.5 Kompetence občanské

Student přijímá názory ostatních spolužáků, svůj názor samostatně prezentuje a opírá ho o potřebné argumenty. Vyjadřuje svůj vlastní názor, posuzuje obě strany v případě konfliktu nad určitým problémem. Při práci ve skupině pomáhá vytvářet a formulovat pravidla potřebná pro fungování skupiny, případně je sám navrhuje. Přijímá důsledky porušení pravidel a v případě konfliktu se na pravidla odvolává.

6.5.6 Kompetence pracovní

Projektové úkoly jsou navrženy tak, aby simulovali problémy skutečného života. Student zvládá plánovat si práci do dílčích segmentů. Je schopný odhadnout čas pro realizaci úkolu, samostatně navrhuje vhodné informační zdroje. Při práci postupuje systematicky, řídí se prezentací, získanými znalostmi, nebo vhodnými informačními zdroji. Svůj postup průběžně vyhodnocuje a v případě potřeby upravuje, aby pracoval efektivně. Student dokáže hodnotit splnění úkolu, ale i úkoly ostatních (na základě stanovených kritérií). Reálně odhaduje své možnosti a dle nich navrhuje časový postup i nutnost nových informačních zdrojů.

6.5.7 Kompetence využívat prostředky IKT a pracovat s informacemi

V každém projektovém úkolu student vyhledává a třídí informace získané z internetu a posuzuje jejich relevantnost. Aktivně využívá počítač a jeho softwarové vybavení (textový editor, grafické programy, editor zdrojového kódu, internetový prohlížeč). Technické řešení webových stránek je hlavní náplní projektové výuky, i navrhnutého kurzu výuky WWW technologií.

6.6 Výukové cíle

Při stanovení výukových cílů jsem vycházel z Bloomovy taxonomie výukových cílů:

1. Zapamatování (popsat, vybrat)
2. Pochopení (zkontrolovat)
3. Aplikace (aplikovat, plánovat, navrhnout, vyzkoušet)
4. Analýza (analyzovat)
5. Syntéza (navrhnout, modifikovat, shrnout)
6. Hodnocení (porovnat, vybrat, zhodnotit)

Projektová výuka má za cíl posilovat schopnosti řešit problémy, technicky myslet, být kreativní, rozvíjet spolupráci a komunikaci s ostatními studenty. Výstupem každého projektového úkolu je textový dokument popisující daný problém, případně konkrétní produkt (webová stránka, nebo její část).

K zapamatování a pochopení slouží teoretická část výuky. V praktické části se zaměřujeme zejména na aplikaci do problémů reálného světa, kde zároveň dochází k upevnění získaných znalostí a jejich prohloubení, pochopení spojitostí. Analýzu provádíme v případě zadání úkolu, kdy jej musíme transformovat do příslušného programovacího jazyka, případně si rozvrhnout šablonu. Naopak syntézu můžeme provést při výměně informací se spolužákem, nebo učitelem, kdy doplňujeme jiný script o vlastní řešení, nebo jej modifikujeme. Hodnocení probíhá formou společné diskuse nad jednotlivými řešeními. Výsledek by měl vždy odpovídat, ale cesty (řešení), jak k němu dojít, mohou být různé. Se studenty doporučuji úkoly konzultovat, diskutovat s celou třídou, jednodušší věci předvést na tabuli, složitější třeba promítnout na projektor.

6.7 Testování získaných znalostí

Pro mě, jako učitele praktického předmětu, jakým informatika bezpochyby je, je velmi důležitá zpětná vazba od studentů. Realizovat výuku jen pro to, abych měl splněné pracovní povinnosti a hodnotit studenty typizovaným, nejhůře jedním testem, je dle mého názoru špatný přístup. Fakt, že si některý student zapamatuje značky pro formátování textu, nebo opíše správné odpovědi od spolužáka, rozhodně neznamená, že je výborný student a bude mu uděleno výborné hodnocení.

Mým cílem je posuzovat studenty komplexně, z toho důvodu doporučuji hodnotit plnění úkolů z lekcí, respektive projektových úkolů. Vyloučíme tak možnost náhody a můžeme znalosti studentů posuzovat komplexně. Několika dotazy k vypracované práci lze velmi snadno zjistit, zda je autorem samotný student, nebo někdo jiný. Součástí jsou také test v polovině kurzu. Při závěrečném hodnocení je nutné zohlednit také přístup studentů, jejich aktivitu při diskusi atp. Při práci na úkolech studenti pracují samostatně, případně ve skupinkách a vyučující se tak může věnovat průběžné kontrole práce a případným dotazům.

6.8 Kompenzum znalostí – potřebné a získané

Pro absolvování kurzu nejsou bezpodmínečně nutné žádné předchozí znalosti z informatiky, ani jiných disciplín. Vítané jsou základní znalosti formátování dokumentu, textových editorů a uživatelská znalost prostředí internetu. Vzhledem k tomu, že se s informatikou studenti setkávají už na základní škole předpokládám, že s výše zmíněným se již seznámili.

Absolvent navrhovaného projektového vyučování by měl disponovat těmito znalostmi:

- Znalost HTML + CSS, základních formátovacích značek, členění dokumentu
- Znalost algoritmizace, schopnost navrhnout a zapsat vlastní algoritmus
- Základní znalost procedurálního PHP
- Schopnost aplikovat algoritmus v praxi, implementovat do jazyka PHP
- Klíčové věci z bezpečnosti na internetu, zásady správného hesla
- Dotazovací jazyk SQL a princip fungování databází
- Základní zabezpečení webu

Vzhledem k faktu, že se v dnešní praxi v oblasti webu vyskytuje velké množství odvětví a již dávno neplatí, že je člověk „tvůrce webu“, navrhoval bych jako navazující dva směry kurzů:

- Frontend vývojář www technologií (kodér)
    - Navazující témata: responzivita, dědičnost stylů, frameworky
- Backend vývojář www technologií (programátor PHP aplikací)
    - Navazující témata: MVC architektura, frameworky

Jako další témata, která již přímo nenavazují na stěžejní témata navrhnutého projektového vyučování, by byly v oblasti SEO optimalizace, UX designu, správy serveru a další.

7 OSNOVA KURZU

Frontální výuka je rozdělena do dílčích časových úseků – lekcí. Lekcí (tedy tematických celků) je celkem 16, přičemž jsou rozděleny do hlavních kategorií:

- Obecné informace v oblasti internetu
- Tvorba šablony (kódování)
- Programování (backend)

Kurz je proložen také jedním testem v jeho polovině. Závěrečný test jsem nezařazoval, druhá polovina kurzu zaměřující se na programování je delší a tak předpokládám, že k hodnocení bude dostatek času i v hodinách. Spousta úkolů je v prezentaci navržena tak, aby se řešila během hodiny, případně studenti odpovídali na otázky v prezentaci, nebo doplňovali vyučujícího při realizaci programu na tabuli. Mimo těchto příležitostí k hodnocení je samozřejmě brán ohled na včas a správně vypracované úkoly, jejichž autorství lze velmi snadno ověřit 1-2 dotazy k hotovému kódu. Dotazy můžeme řešit také formou diskuse, nebo dialogu dvou žáků s přihlížením vyučujícího.

7.1 Lekce 1 – Úvod do WWW technologií

V první lekci je důležité zjistit pre-koncepty a mis-koncepty studentů a jejím cílem je sjednotit jejich znalosti na stejnou úroveň. Lekce je tedy spíše teoretická a týká se základních informací ohledně fungování internetu, zejména:

- Historie Internetu
- Funkce Internetu a internetu
- Pojmy související s internetem a weby
- Webový prohlížeč, funkce serveru
- Statické x dynamické webové stránky

Někteří studenti mají velmi mylné představy o Internetu. Pod tímto pojmem si představují konkrétní web, např. seznam.cz a vůbec netuší jak vlastně Internet funguje. Závěrem lekce je připravena jednoduchá hra, kterou si ověříme sjednocení úrovně nad určitou minimální mez. Téma této lekce je součástí projektové výuky.

Časová náročnost: 2 vyučovací hodiny (2x45min)

Časový rozvrh 2+0 (2h přednáška)

7.2 Lekce 2 – Bezpečnost na internetu

S rozvojem internetových aplikací v mnoha zařízeních a neustále zvyšujícímu se počtu zařízení, které lze prostřednictvím internetu ovládat, by mělo růst i povědomí o bezpečnosti na internetu. Nedílnou součástí autentizace uživatele je zadávání hesla. Studenty ve zkratce seznámíme s vhodnými hesly a nastíníme jim i možnosti zabezpečení z pohledu správce webu, programátora. Cílem přednášky je studenty seznámit s možnými riziky a osvětlit jim problematiku autentizace a hesel, zejména v prostředí internetu, kde jsou používána nejčastěji. Téma této lekce je součástí projektové výuky.

Časová náročnost: 1 vyučovací hodina (1x45 minut)

Časový rozvrh 1+0 (1h přednáška)

7.3 Lekce 3 – Tvorba šablony – obecně

V této lekci se dostáváme k praktické části kurzu a když už z předchozích kurzů víme co webová stránka je a jak funguje, zkusíme si vytvořit svou vlastní. Tato lekce je jakýmsi úvodem do vytvoření šablony, studentům sdělíme zásady při tvorbě webu, vysvětlení pojmů (tag, deklarace typu dokumentu, tělo, hlavička atp.), práci s vývojovým prostředím, ukládání na FTP server atd. Výstupem bude vlastní HTML soubor, stránka „Ahoj světe“ s použitím několika málo základních tagů.

Cíle:

- Zapamatování si základních konstrukcí jazyka HTML
- Pochopení funkce tagů a atributů v html dokumentu
- Pochopení zanořování tagů

Časová náročnost: 2 vyučovací hodiny (2x45 minut)

Časový rozvrh: 1+1 (1h přednáška, 1h cvičení)

7.4 Lekce 4 – Tvorba šablony – HTML

Z předchozí lekce již studenti získali základ potřebný k vytvoření vlastního „webu“. Nyní je třeba procvičit si značkovací jazyk HTML, seznámit se z jednotlivými tagy a jejich funkcemi.

Cíle:

- Osvojení si základní struktury webového dokumentu
- Pochopení rozdílů mezi párovými a nepárovými tagy
- Vytvoření vlastní webové stránky
- Zapamatování si vlastností používaných tagů

Časová náročnost: 2 vyučovací hodiny (2x45min)

Časový rozvrh: 1+1 (1h přednáška, 1h cvičení)

7.5 Lekce 5 – Tvorba šablony – CSS

Do připravené šablony zkusíme aplikovat kaskádové styly. Studenti si vyzkouší měnit CSS tak, aby web přizpůsobili své představě a naučili se:

- Změnit typ písma (fontu), barvu a velikost
- Stylovat různé prvky (implicitní h1, p, span, div)
- Stylovat třídy a identifikátory
- Rozdíly mezi inline, interním a externím zápisem CSS
- Základní pozicování

Cíle:

- Pochopení propojení kaskádových stylů s HTML
- Zapamatování si konstrukcí jazyka CSS
- Pochopení dědičnosti v CSS
- Zapamatování si vybraných vlastností, které vycházejí s AJ

Časová náročnost: 2 vyučovací hodiny (45min) + domácí úkol

Časový rozvrh: 1+1

7.6 Lekce 6 – Vytvoření vlastního webu (shrnutí)

V této lekci shrneme vědomosti z předchozích kurzů a zkusíme vytvořit svůj druhý web, který už nebude jednoduchou stránkou typu „wordovský dokument“, ale bude se malinko více podobat klasickému webu. Vyzkoušíme si pozicování pomocí float a vytvoříme si tak klasickou stránku v podobě, jak ji známe z Internetu (rozdělenou na menu a „content“, obsah, záhlaví a zápatí pro zjednodušení vynecháme). Doplníme si znalosti z hlavičky dokumentu, vč. meta tagů pro kódování.
Vzhledem k tomu, že je pro zapamatování si všech tagů potřeba delší doba než jsme tématu dosud věnovali, mají studenti k dispozici veškeré zdroje, Internet, rady vyučujícího. Programování (v tomto případě kódování), je také o tom, umět si dohledat relevantní informace, případně spolupracovat v týmu. Z tohoto důvodu považuji za nesmyslné vyžadovat po studentech exaktní znalosti určitého množství tagů. Důležité je, aby studenti kód zpracovali sami, rozuměli mu a dokázali jej vysvětlit, nebo doplnit. Tím se ověří jejich znalosti ze samostatné práce.
Cíle:

- Upevnit získané znalostí z předchozích lekcí
- Rozvinout schopnost spolupráce (konzultace s ostatními)
- Ověřit schopnost dohledat si relevantní informaci na internetu

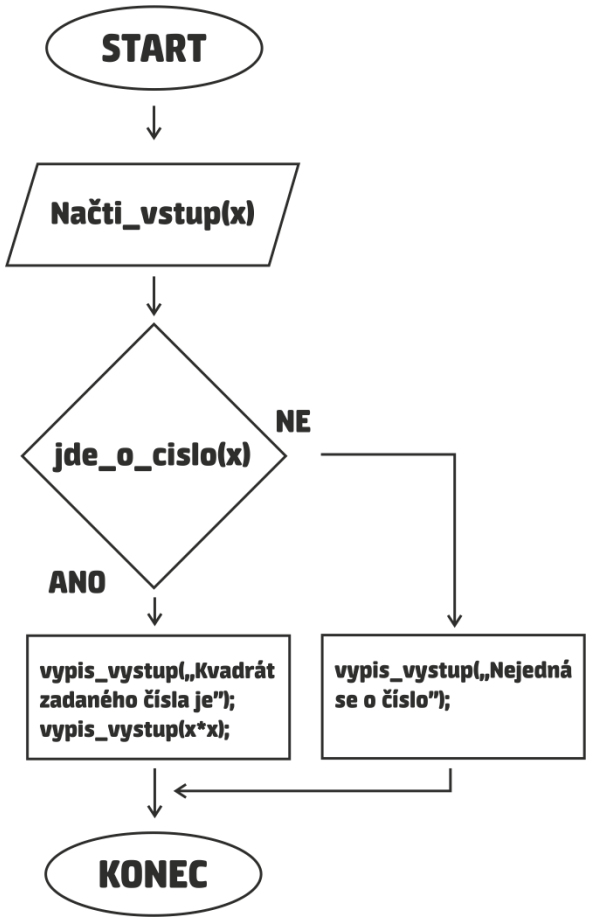
Časová náročnost: 4 vyučovací hodiny (4x45 minut)
Časový rozvrh: 1+3 (1h přednáška, 3h cvičení)

7.7 Test

Je na zvážení každého vyučujícího, zda využije tento půl-semestrální test. Shrnuje dosažené znalosti v obecných otázkách a může sloužit jako jedna z částí pro finální hodnocení. Společně s předchozím samostatným projektem bude klíčovým hodnocením této 1. poloviny kurzu zabývající se tvorbou šablony.
Test je koncipován tak, aby ověřil porozumění základním HTML a CSS kódům a jejich logické návaznosti.
Časová náročnost: 1h

7.8 Lekce 8 – Základy algoritmizace

K programování neodmyslitelně patří algoritmizace. Jelikož je náš kurz pouze základní, bude mít především za cíl sjednotit úroveň studentů nad minimální hranici, podmiňující zvládnutí následujících lekcí. V této lekci si probereme, co algoritmus znamená, jak se zapíše, jeho vazbu na programovací jazyk a základní struktury algoritmů. Datové struktury a vícenásobné, vnořené cykly nebudou obsahem naší lekce a mohly by navazovat na náš kurz. Výuka je teoretická, potřebovat budeme pouze projektor a psací potřeby.



Obrázek 11 – Vývojový diagram

Cíle:

- Prohloubit analytické myšlení (schopnost analyzovat daný problém)
- Získat schopnost vytvářet vývojové diagramy
- Upevnit si schopnost samostatné a zodpovědné práce
- Ověřit schopnost obhájit si své vlastní řešení a diskutovat o něm

Časová náročnost 6 vyučovacích hodin (6x45 minut)

Časový rozvrh: 4+2 (přednáška kombinovaná se cvičením v poměru 2:1)

7.9 Lekce 9 - Úvod do PHP

V této lekci by se studenti měli seznámit s jazykem PHP, s jeho dynamickým typovým systémem, základními konstrukcemi a spouštěním vlastní aplikace na serveru v reálném čase.

Cíle:

- Pochopení problematiky datových typů
- Obeznámení s matematickými operacemi a z řetězením textů v jazyce PHP
- Podpořit kreativitu studentů

Časová náročnost 2 vyučovací hodiny (2x45 minut)

Časový rozvrh: 1+1 (přednáška kombinovaná se cvičením v poměru 1:1)

7.10 Lekce 10 – První aplikace v PHP (kalkulačka)

Studenti si vyzkouší vytvořit jednoduchou kalkulačku, která po zadání dvou čísel provede jejich součet, součin, rozdíl a podíl. Nejdříve si vyzkouší hodnoty zadat přímo do programu, jako další krok bude následovat zpracování vstupů z formuláře.

Cíle:

- Procvičit problematiku řetězení
- Početní operace s proměnnými
- Typový systém a proměnné
- Opakování HTML formulářů

Časová náročnost: 2 vyučovací hodiny (1x45 minut)

Časový rozvrh 1+1

7.11 Lekce 11 – Větvení v PHP

Připomeneme studentům lekci algoritmizace a ukážeme si praktické propojení s praxí na příkladu naší kalkulačky. Větvení využijeme k rozhodování, jakou operaci uživatel zvolil. Přiblížíme se tak skutečné verzi kalkulačky, ve které nechceme nutně provádět všechny operace, ale pouze některé.

Vytvoříme si také formulář pro vkládání uživatelů a podle věku (pomocí větvení) každému přiřadíme kategorii, dítě, dospělý, senior.

Cíle:

- Zopakovat problematiku HTML formulářů
- Upevnit si problematiku větvení v PHP (analogie z algoritmizace)
- Ověřit schopnost samostatné práce

Časová náročnost: 2 vyučovací hodiny (1x45 minut)

Časový rozvrh 1+1

7.12 Lekce 12 – Cykly

V lekci zaměřené na cykly si připomeneme princip cyklů z algoritmizace, studentům představíme taxonomii cyklů a naučíme se jejich správné použití na příkladech. Součástí přednášky jsou jednoduché úlohy, které řešíme společně se studenty na tabuli. V závěru lekce si představíme speciální typ cyklu používaný v PHP „foreach“ a nastíníme problematiku vnořených cyklů.

Cíle:

- Ujasnění rozdílů mezi jednotlivými PHP cykly
- Pochopení výhod použití cyklů
- Zopakování algoritmizace – vývojových diagramů
- Pochopení problematiky vnitřního (vnořeného) cyklu

Časová náročnost: 2 vyučovací hodiny (1x45 minut)

Časový rozvrh 1+1

7.13 Lekce 13 – Práce s poli

V této lekci si definujeme pojem „pole“ v oblasti datových struktur v informatice a naučíme se jej používat. Po procvičení na tabuli studentům představíme vícerozměrná pole a ukážeme si jejich praktické využití. V závěru hodiny si představíme asocitativní pole a v rámci úkolu si všechny své získané znalosti studenti upevní.

Cíle:

- Procvičení problematiky indexovaných polí v PHP
- Procvičení problematiky asociativních polí v PHP
- Upevnění znalostí z přednášky na základě aplikace do vlastní úlohy
- Zopakování cyklů

Časová náročnost: 3 vyučovací hodiny (1x45 minut)

Časový rozvrh 1+2

7.14 Lekce 14 – Vytvoření databáze

Databáze, společně s PHP tvoří nedílnou součást při vývoji dnešních webových stránek. Data je nutné někam ukládat a jako nejefektivnější způsob jsou i pro menší aplikace vhodné databáze. Nejpoužívanějším typem je databáze mySQL, se kterou se studenti setkají i v praxi. Studenti se během lekce naučí používat nástroj pro správu databáze (phpMyAdmin) v jeho grafické podobě, vytvořit, editovat a mazat databázovou tabulkou. Seznámí se také s datovými typy pro databáze a naučí se je používat. Představíme si dotazovací jazyk SQL, který využijeme v další lekci. V projektovém úkolu si studenti během samostatné činnosti ověří a upevní získané znalosti.

Cíle:

- Naučit se používat nástroj pro správu databáze
- Pochopit strukturu databáze
- Naučit se vytvořit v grafickém prostředí databázovou tabulku
- Osvojit si plnění tabulek v databázi daty
- Pochopit problematiku datových typů

Časová náročnost: 3 vyučovací hodiny (3x45 minut)

Časový rozvrh 2+1

7.15 Lekce 15 – Implementace databáze do aplikace

Tato lekce přímo navazuje na lekci č. 14, úvod do databází. Studenti umí databázi vytvořit a naplnit daty pomocí nástroje pro správu databáze. V této lekci se naučíme databázi propojit s naším PHP scriptem a provádět její „vzdálenou“ obsluhu přímo v našem scriptu. Studenti využijí hotový kód, kterému porozumí a dokáží jej modifikovat pro své potřeby.

Cíle:

- Pochopit provázání databáze a PHP scriptu
- Naučit se použití scriptu pro výpis dat
- Naučit se použití scriptu pro vkládání dat
- Zopakování jazyka SQL
- Ověření schopnosti dohledat si nové informace (příkazy SQL jazyka)
- Zvýšení schopnosti orientace v cizím kódu
- Schopnost analýzy cizího kódu a jeho pochopení
- Prohloubení práce s komentáři

Časová náročnost: 3 vyučovací hodiny (3x45 minut)

Časový rozvrh 1+2

7.16 Lekce 16 – Zabezpečení PHP aplikace

V posledním úkolu se studenti seznámí se základním zabezpečením databáze a jeho jednodušší variantu se pokusí aplikovat na svoji aplikaci, tak aby byla zabezpečená, alespoň na minimální úrovni. Jako dobrovolný úkol je připravená také varianta s zabezpečením proti SQL injection pomocí tzv. „prepared statements“.

Cíle:

- Naučit se základní zabezpečení aplikace
- Pochopit význam použitých funkcí
- Seznámit se s metodou zabezpečení pomocí prepared statements

Časová náročnost: 2 vyučovací hodiny (2x45 minut)

Časový rozvrh 1+1

8 PROJEKTOVÉ ÚKOLY

Projektová výuka byla navrhnuta jako alternativa ke klasické výuce, resp. k jejímu doplnění. Každý student je individuální a každému vyhovuje jiná výuková metoda a forma. Projektová výuka učí studenty dovednosti rozvrhnout si vlastní možnosti a čas, podporuje jejich kreativitu, rozvíjí týmovou spolupráci a připravuje studenty pro budoucí praxi, zaměstnání.

8.1 Úvod do tvorby WWW stránek

Projektový úkol je navrhnut jako alternativa k lekci č. 1 a seznamuje studenty s prostředím webových stránek z pohledu tvůrce webu. Zaměřuje se na možnosti tvorby webu, základy internetu (IP adresa, doménové jméno, pojmy v oblasti WWW technologií) a obory zabývající se tvorbou webových stránek.

Výstupem je prezentace svých výsledků a diskuse nad nimi s ostatními spolužáky a vyučujícím.

K tomuto úkolu je připojeno i vzorové řešení, které může být jednak použito vyučujícím k pochopení mého záměru a také může být prezentováno studentům po splnění tohoto úkolu pro doplnění, případně nasměrování, aby v dalších úkolech měli jasno, jakým způsobem pokračovat a úkoly plnit.

8.1.1 Cíle projektového úkolu

- Seznámit se s možnostmi tvorby webových stránek
- Upevňuje/rozšiřuje znalosti v oblasti internetu (domény, IP adresy, pracovní pozice při tvorbě WWW stránek)
- Ověřit schopnost pracovat samostatně a vytvořit ucelenou výstupní zprávu (v praxi např. pro nadřízeného)
- Ověřit schopnost samostatně prezentovat výsledky své práce

8.1.2 Výstupy projektového úkolu

- Komplexní zpráva na téma tvorby webových stránek a souvisejících pojmů
- Prezentace

8.2 Je na internetu bezpečno?

Přes obrovský pokrok v oblasti kybernetické bezpečnosti je stále Internet místo, kde dochází k masivnímu přenosu dat, Internet využívají miliardy lidí na celém světě a je tak velmi lákavým místem pro útočníky. Nejenže přibývá samotných uživatelů Internetu, ale přibývá také množství času, které na něm uživatelé tráví a s tím je spojeno větší množství používaných aplikací, které lze napadnout. Studenti by se během tohoto projektového úkolu měli zamyslet nejen nad vlastním chováním na Internetu z pohledu uživatele, ale jako tvůrci webových stránek také nad chováním ostatních uživatelů a možnostech jejich ochrany.

8.2.1 Cíle projektového úkolu

- Uvědomit si možné hrozby při používání Internetu (afektivní cíl)
- Získat povědomí o možnostech ochrany na Internetu
- Rozebrat pojem „bezpečného hesla“
- Rozvíjet týmovou spolupráci (sociální cíl)

8.2.2 Výstupy projektového úkolu

- Ve dvojicích vypracovaná finální zpráva o bezpečnosti na internetu
- Prezentace získaných výsledků své práce před třídou

8.3 Tvorba šablony webu (frontend)

Tento projektový úkol si klade za cíl vytvořit šablonu reálné webové stránky, která bude syntakticky i sémanticky správná. Téma webové stránky je libovolné, studenti teda mají prostor pro svou kreativitu a můžou navrhnout úvodní stránku svého blogu, nové stránky školy, cokoliv je napadne.

8.3.1 Cíle projektového úkolu

- Ucelit si znalosti z tvorby šablony WWW stránky
- Syntetizovat jednotlivé části tvorby webu a vytvořit kompletní webovou stránku
- Rozvíjet schopnosti týmové práce, komunikační schopnosti (sociální cíl)
- Ověření schopnosti dohledat si relevantní informace na Internetu

- Vytýčení a přijetí sociálních rolí v týmu (sociální cíl)

8.3.2 Výstupy projektového úkolu

- Kompletní webová stránka
- Časový plán tvorby WWW stránky
- Společná prezentace shrnující postup a splnění cílů

8.4 Poznáváme algoritmy

Projektový úkol, jehož hlavním tématem jsou základy algorimizace je stěžejním tématem před samotným programováním. Je dobré udržet si jistou míru abstrakce a naučit se nejdříve myslet „jako počítač“ a až následně interpretovat svůj záměr v konkrétním jazyce. Z toho důvodu je tento projektový úkol zaměřen na pochopení programování bez konkrétního jazyka. Vývojové diagramy jsou také vhodné pro začátečníky, pokud při převodu do programovacího jazyka dojde k nějakému sémantickému problému, je pro druhého programátora, resp. učitele snadné vysvětlit studentovi daný problém a program opravit.

8.4.1 Cíle projektového úkolu

- Rozvoj algoritmického, systematického myšlení
- Naučit se formulovat problémy pomocí vývojového diagramu
- Rozvoj tvořivosti studentů
- Podpořit dovednost analyzovat a řešit konkrétní problémy (úlohy)

8.4.2 Výstupy projektového úkolu

- Dokument popisující algoritmizaci
- 3 úlohy řešené pomocí vývojového diagramu

8.5 Programujeme webovou aplikaci – PHP formulář

Programování vlastní webové aplikace, tedy scriptu, který obsluhuje webovou stránku zevnitř a tvoří tak dynamickou část, bez které se drtivá většina webů, internetových obchodů neobejde považuji za jakýsi vrchol našeho kurzu. Studenti se v tomto projektovém úkolu zaměří na formulář, který dokáži zpracovat, výstup editovat, tisknout na obrazovku a případně odesílat jako zprávu na zadaný email.

8.5.1 Cíle projektového úkolu

- Procvičení problematiky indexovaných polí v PHP
- Procvičení problematiky asociativních polí v PHP
- Upevnění znalostí na základě aplikace do vlastní úlohy
- Zopakování cyklů
- Ověření znalostí z tvorby šablony (HTML + CSS)
- Podpořit kreativitu a vlastní rozhodování, vč. zodpovědnosti za výsledný produkt

8.5.2 Výstupy projektového úkolu

- šablona obsahující nastylovaný webový formulář
- PHP script tvořící výstup z formuláře, odesílání na email
- šablona pro zobrazení výstupních dat

8.6 Pracujeme s databází

K modernímu webu, ve spojitosti s dynamickou webovou stránkou neodmyslitelně patří databáze. Obsah je potřeba aktualizovat, mít ho dobře formátován, potřebujeme pracovat s jednotlivými buňkami, ale i celými tabulkami. To při ukládání obsahu do souboru není možné a vytvořit zázemí např. internetového obchodu s ukládáním do souborů by bylo velmi nepřehledné, složité a neefektivní z hlediska prostorové, i časové složitosti. Z toho důvodu se programátor webových aplikací neobejde bez databází. V projektu zaměřeném na databáze se studenti seznámí s základními příkazy a operacemi, které lze s databázemi provádět. Vyzkouší si vytvořit vlastní tabulky a naplnit je testovacími daty, jak v grafickém režimu, tak zápisem SQL kódu. Naučí se vložená data vypsat s různými podmínkami, setříděná atp.

8.6.1 Cíle projektového úkolu

- Porozumět grafickému prostředí nástroje pro správu databáze
- Dokázat pracovat s nástrojem pro správu databáze (přidávat, editovat a mazat data)
- Pochopit problematiku datových typů a umět aplikovat správný datový typ na položku
- Podporovat kreativitu studentů při volbě vlastního tématu práce, i obsahu celé databáze

8.6.2 Výstupy projektového úkolu

- Vytvořené databázové tabulky naplněné testovacími daty
- Prezentace vlastní databáze s komentářem

9 REALIZACE VÝUKY NA SŠ

Reflexi od studentů považuji za velmi důležitou a měl jsem v plánu kurz upravit se zapracováním jejich poznámek. Pochopitelně jedna třída (2 skupiny) nejsou příliš objektivní vzorek. Pro řádné otestování bych musel výuku vyzkoušet minimálně na 10 skupinách. Nicméně i tak jsem předpokládal, že některé poznámky zvážím a kurz upravím. Vzhledem ke vzniku pandemie na území ČR v souvislosti s virovým onemocněním COVID-19 a plošnému uzavření škol k 11.3. jsem měl možnost absolvovat pouze jeden termín výuky v rámci 2 skupin. Původně bylo plánováno 10 hodin výuky (jedna přednáška + projektový úkol č. 3).

Výuku jsem realizoval na Obchodní akademii v Šumperku ve druhém ročníku. První téma (a vzhledem k pandemii i jediné), které mi přišlo velmi důležité jsem zvolil „Bezpečnost na internetu“. Studenty prezentace zaujala a dle zpětné vazby byla i přínosná. Jedna ze studentek se vyjádřila ve smyslu, že se jí zpracování prezentace líbí, je přehledné, že je informací rozumné množství a je zajímavá. I z chování studentů v hodině usuzuji, že byla prezentace dostatečně zajímavá, z pozorování si dovolím říci, že 95% času věnovali mému výkladu a prezentaci. K realizaci vybraného projektového úkolu (č. 3) již bohužel nedošlo.

Součástí byl krátký test, abych si ověřil, že studenti opravdu dávali pozor a snažili si učivo zapamatovat (test je součástí práce v příloze). Výsledky testů měly úspěšnost 78%, což považuji za uspokojivý výsledek. Na realizaci projektového úkolu už z výše uvedených důvodů nedošlo.

ZÁVĚR

V teoretické části práce byli popsány technologie používané k tvorbě webových stránek, formy výuky na školách, RVP pro tvorbu WWW stránek a podstatná část práce je věnovaná také bezpečnosti na internetu. Práce k tématům přistupuje obecně, popisuje jednotlivé technologie a potřebné nástroje k realizaci výuky. Rozebírá programové vybavení, vlastnosti používaných jazyků, formy výuky, řeší databáze, problematiku bezpečného hesla a další. Za důležitý považuji, mimo samotný obsah výuky a její formu, také přístup vyučujícího. Z toho důvodu se jedna pasáž věnuje i pedagogickým zásadám.

Praktická část se zabývá realizací výuky WWW technologií na SŠ a návrhem pracovních listů pro výuku. Z důvodu komplexnosti je navrhnuta frontální i projektová výuka. V práci jsou rozebrány jednotlivá témata kurzu, důvod jejich zvolení, časová náročnost, mezipředmětové vztahy, plnění klíčových kompetencí a naplnění bodů Bloomovy taxonomie. Součástí práce jsou přílohy, které obsahují navrhnutou výuku (16 lekcí), z nichž každá (mimo testu) obsahuje prezentaci a případně samostatnou práci (někdy řešenou i v prezentaci v rámci vyučovací hodiny). K důležitým úkolům jsou přiložena i řešení pro demonstraci studentům, případně pro doplnění znalostí vyučujícího. Další příloha obsahuje 6 projektových úkolů, které mají za cíl doplnit, případě nahradit některé části výuky. Projektová výuka může plně nahradit klasickou výuku, nicméně doporučuji zkombinovat obě možnosti. Součástí práce je také zpětná vazba od studentů, bohužel vzhledem k plošnému uzavření škol pouze z 2 vyučovacích hodin. V rámci reflexe z výuky jsem obdržel kladné ohlasy a studenti dosáhli v krátkém testu v závěru hodiny 78% úspěšnosti v zapamatování si některých částí prezentace. Celkově výuku hodnotili jako přínosnou a obsahově velmi hodnotnou. Je zde předpoklad, že při realizaci celé výuky na konkrétní škole by došlo ke zpřesnění požadované časové dotace a úpravě některých prezentací z hlediska časového rozložení. Eventuelně změna některých úkolů, přizpůsobení dané škole (z hlediska oboru, i genderu). Celkově práci považuji za hotovou a jednotlivé úkoly i celý kurz připravý k realizaci výuky webových technologií na SŠ.

SEZNAM POUŽITÉ LITERATURY

[1] **PRŮCHA, J., WALTEROVÁ, E., MAREŠ, J. a kol.** *Pedagogický slovník, 4. vydání.* Praha : Portál, 2003.

[2] **KRATOCHVÍLOVÁ, Jana.** *Teorie a praxe projektové výuky. 2. vydání.* Brno : Masarykova univerzita, 2016.

[3] **Maňák, Josef a Švec, Vlastimil.** *Výukové metody.* Brno : Paido, edice pedagogické literatury, 2003.

[4] **Komenský, Jan Amos.** *Analytická didaktika.* Praha : Státní nakladatelství, 1947.

[5] **MAREŠ, J. OUHRABKA, M.** Žákovo pojetí učiva. Pedagogika: Časopis pro vědy o vzdělávání a výchově. [Online] 1992. https://pages.pedf.cuni.cz/pedagogika/?p=3566&lang=cs.

[6] **Šenkyřík, Daniel.** Seznámení a konfigurace s Apache. *Interval.cz.* [Online] 30. 07 2015. https://www.interval.cz/clanky/seznameni-a-konfigurace-s-apache/.

[7] **Fiala, Jan.** Editor PSPad. *Eitor PSPad.* [Online] 15. 01 2020. http://pspad.com.

[8] **Hogan., Brian P.** *HTML5 a CSS3. Výukový kurz webového vývojáře.* Brno : Computer Press, 2012.

[9] **Ulman, Larry.** *PHP a MySQL.* Brno : Computer Press, 2004.

[10] **Čápka, David.** Úvod do ASP.NET. *Interval.cz.* [Online] 13. 11 2012. https://www.itnetwork.cz/csharp/asp-net/tutorial-uvod-do-asp-dot-net.

[11] **Gilmore, W.J.** *Velká kniha PHP 5 a MySQL: kompendium znalostí pro začátečníky i profesionály. Nové 3. vyd. Přeložil Jan Pokorný.* Brno : Zoner press, 2011. 978-80-7413-163-9.

[12] **Vrána, Jakub.** Správa databáze v jednom PHP souboru. *Adminer.* [Online] 04. 05 2020. https://www.adminer.org/cs/.

[13] **MITCHELL, R.** *Open source CMS: Zvolte si nejvhodnější.* Praha : Computerworld, 2013.

[14] **Engebretson, Patrick.** *The Basic of Hacking and Penetration Testing: Ethical Hacking and Penetration Testing Made Easy.* místo neznámé : Syngress, 2011. 9780124116443.

[15] **Gao, W. a Kim, J.** *Robbing the cradle is like takingcandy from a baby. Proceedings of the Annual Conference of the Security Policy Institute (GCSPI).* Amsterdam : autor neznámý, 2007. 1.23-37.

[16] **Mitnick, K. a Simon,W.L.** *Umění klamu.* místo neznámé : Helion, 2002. 978-83-7361-210-5.

[17] Hashovací funkce. *Mendelova Univerzita v Brně.* [Online] https://is.mendelu.cz/eknihovna/opory/zobraz_cast.pl?cast=7029.

[18] Solení hesel aneb Sůl nad zlato. *PHP Guru.* [Online] Jan Tichý, 30. 10 2007. [Citace: 6. 12 2019.] https://www.phpguru.cz/clanky/soleni-hesel.

[19] **MŠMT.** RVP pro Informační technologie. *Národní ústav pro vzdělávání.* [Online] 29. 05.2008. http://zpd.nuov.cz/RVP/ML/RVP%201820M01%20Informacni%20technologie.pdf.

SEZNAM POUŽITÝCH SYMBOLŮ A ZKRATEK

URL	Uniform Resource Locator
HTML	HyperText Markup Language
CSS	Cascading Style Sheets
PHP	HyperText Preprocesor
IDE	Integrated Development Environment (vývojové prostředí)
SQL	Structured Query Language (dotazovací jazyk pro databáze)
MVC	Model-view-controller (architektonický vzor)
Frontend	Část webu, kterou vidí každý návštěvník stránek, šablona webu
Backend	Programová část webu (obsluha webu)
RVP	Rámcový vzdělávací program
ŠVP	Školní vzdělávací program
MŠMT	Ministerstvo školství, mládeže a tělovýchovy
WYSWYG	What you see, it what you get (editor HTML tagů)
Opensource	Volně šířitelný software s otevřeným zdrojovým kódem
CMS	Content managet systém (v kontextu systém pro správu WWW stránek)
RS	Redakční systém (totožné s CMS)

SEZNAM OBRÁZKŮ

SEZNAM TABULEK

SEZNAM PŘÍLOH