Česká zemědělská univerzita v Praze

Provozně ekonomická fakulta

Katedra informačního inženýrství

Bakalářská práce

Webové stránky obce

Martin Kolářík

ČESKÁ ZEMĚDĚLSKÁ UNIVERZITA V PRAZE

Provozně ekonomická fakulta

ZADÁNÍ BAKALÁŘSKÉ PRÁCE

Martin Kolářík

Informatika

Název práce

Webové stránky obce.

Název anglicky

Municipal Government Websites.

Cíle práce

Cílem této práce je navrhnout a vytvořit webové stránky obce, na základě analýzy potřeb dané obce i osob navštěvujících tyto stránky.Vývoj stránek bude založen na platformě C#.net.

Metodika

Na základě výsledků analýzy, bude v online editoru Gomockingbird udělán logický návrh stránek. Pomocí tohoto návrhu vytvořím v jazyce C#.net, za použití programu Visual Studio, funkční webové stránky. Vývoj stránek bude založen na Bootstrap frameworku a dalších veřejně dostupných nástrojích.Pro ukládání a správu dat bude využit Microsoft SQL server.

Doporučený rozsah práce

30-40 stran

Klíčová slova

Webové stránky obce, C#, Bootstrap framework

Doporučené zdroje informací

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

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

PÍSEK, S. *ASP.NET : začínáme programovat : podrobný průvodce začínajícího uživatele.* Praha: Grada, 2003. ISBN 80-247-0526-5.

SHELDON, R. *SQL : začínáme programovat.* Praha: Grada, 2005. ISBN 80-247-0999-6.

Předběžný termín obhajoby

2017/18 LS – PEF

Vedoucí práce

Ing. Marek Pícka, Ph.D.

Garantující pracoviště

Katedra informačního inženýrství

Elektronicky schváleno dne 11. 1. 2018

Ing. Martin Pelikán, Ph.D.

Vedoucí katedry

Elektronicky schváleno dne 11. 1. 2018

Ing. Martin Pelikán, Ph.D.

Děkan

V Praze dne 09. 03. 2018

Čestné prohlášení

Prohlašuji, že svou bakalářskou práci "Webové stránky obce" jsem vypracoval(a) samostatně pod vedením vedoucího bakalářské práce a s použitím odborné literatury a dalších informačních zdrojů, které jsou citovány v práci a uvedeny v seznamu použitých zdrojů na konci práce. Jako autor(ka) uvedené bakalářské práce dále prohlašuji, že jsem v souvislosti s jejím vytvořením neporušil autorská práva třetích osob.

V Praze dne 15.3.2018 _____

Poděkování

Rád bych touto cestou poděkoval panu Ing. Marek Pícka, Ph.D. za cenné rady a odborné vedení při zpracovávání této práce.

Webové stránky obce

Abstrakt

V teoretické části jsou uvedeny souhrnné vlastnosti a způsob práce s programovacím jazykem C#.net. Dále jsou zde popsány ostatní technologie, které jsou potřebné při vytváření webu, jako jsou například CSS styly, Java Script, jQuery a další. Na konci teoretické části jsou definovány použité frameworky, jejich výhody a nevýhody. Také jsou zde určeny způsoby, pomocí kterých lze publikovat vytvořené stránky na internetové síti.

V praktické části je na základě provedené analýzy, vytvořen logický návrh webové stránky a definovány potřebné funkcionality. Dále je zde uvedeno praktické využití programovacích jazyků a jejich propojení s frameworky tak, aby vznikl jeden funkční celek odpovídající logickému návrhu. Poté je vytvořen grafický návrh, podle kterého se stránky upraví do konečného stavu.

Klíčová slova: webové stránky, Visual studio, C#.NET, MVC, Bootstrap, CSS, jQuery, MSSQL

Municipal Government Websites

Abstract

In the theoretical part are presented the summary features and way of working with the C#.NET programming language. Next, there are described other technologies which are needed when creating a site, such as CSS styles, Java Script, jQuery, and more. At the end of the theoretical part, there are defined property of used frameworks, their advantages and disadvantages. There are also ways to publish the created pages on the internet network.

In the practical part is created the logical design of the website based on analytic results and defined needed functionality. Next here is he the practical use of programming languages and their connection with frameworks to be created one working unit corresponding with structural level. At last is created graphical design and final version of websites based on this graphic design.

Keywords: web sites, Visual studio, C#.NET, MVC, Bootstrap, CSS, jQuery, MSSQL

Obsah

Seznam obrázků

Seznam použitých zkratek

AJAX	Asynchronous JavaScript and XML
CSS	Cascading Style Sheets
HTML	HyperText Markup Language
IIS	Internet Information Services
MSSQL	Microsoft Structured Query Language Server
SQL	Structured Query Language
XML	eXtensible Markup Language

1 Úvod

V dnešní době vlastní už téměř každý nějaké zařízení, pomocí kterého se lze připojit k internetu. Internet nám poskytuje důležité informace a šetří nám čas, který bychom jinak vynaložili na jejich vyhledávání.

Navzdory tomuto faktu je stále velké množství obcí, které své webové stránky neaktualizují nebo na nich neposkytují potřebné informace a funkce, které obyvatelé vyžadují. Také se velmi často stává, že jsou stránky optimalizované pouze pro zařízení s velkou zobrazovací plochou, jako jsou desktopové počítače a například na mobilních telefonech je jejich prohlížení velmi problematické.

Cílem mé práce bude projít celým cyklem vytváření webových stránek od analýzy, logického návrhu až po testování a úplné nasazení. A takto vytvořit stránky, které budou odpovídat dnešním standardům. Je kladen důraz na to, aby obsah webu mohl kdokoliv upravovat a aktualizovat i bez znalostí programovacích jazyků.

2 Cíl práce a metodika

2.1 Cíl práce

Cílem této práce je navrhnout a vytvořit funkční stránky obce. Stránky budou vytvořené v jazyce C# za pomocí .NET frameworku a MVC návrhu. Obsah stránky musí být možné upravovat a aktualizovat online, pomocí webového prohlížeče, a to i osobám bez znalosti jakéhokoliv programovacího jazyka. Také musí být implementován systém přístupových práv. Stránky budou optimalizované pro klasické stolní počítače i mobilní zařízení.

2.2 Metodika

Na základě výsledků analýzy bude v online editoru Gomockingbird udělán logický návrh stránek. Pomocí tohoto návrhu vytvořím v jazyce C#.NET, za použití programu Visual Studio, funkční webové stránky. Vzhled stránky bude vytvořen pomocí Bootstrap frameworku a CSS stylů. Data budou ukládaná do MSSQL databáze a přístup k nim bude zajištěn Entity Frameworkem. Zabezpečení a přístupová práva stránky obstará Identiy Framework. Nakonec budou stránky publikovány na serveru Microsoft Azure.

3 Teoretická východiska

3.1 Webová stránka

Webová stránka nejčastěji představuje dokumenty uložené na nějakém serveru, který je připojen do internetové sítě. Každá stránka je jednoznačně identifikována pomocí své URL adresy. Základní technologií pro vytváření stránek je HTML jazyk. Jednotlivé stránky jsou interpretovány pomocí webových prohlížečů, které nám zobrazují výslednou stránku na základě jejího zdrojového kódu. To znamená, že jsou takzvaně multiplatformní, tedy že nezáleží na operačním systému zařízení, ale pouze na webovém prohlížeči, ve kterém jsou zobrazeny. [1]

3.1.1 HTML

Název HTML vychází z anglického „HyperText Markup Language“ neboli jazyk značek a odkazů. Jedná se o značkovací jazyk, kde je pomocí jednotlivých značek vytvářen obsah stránky. Jednotlivé značky určují pouze jaké budou prvky na stránce a jaký bude jejich obsah. Prvky mohou být například tabulky, obrázky, nadpisy, odstavce, seznamy a mnoho dalších. HTML bylo původně určeno i pro grafickou úpravu stránky, ale postupným vývojem se od toho upustilo a používají se k tomu jiné metody jako například CSS. [2]

Pohyb mezi jednotlivými stránkami je zajištěn pomoci hypertextových odkazů. Ty mohou být dvojího typu. První typ slouží pro pohyb mezi jednotlivými dokumenty. Druhý typ slouží pro pohyb v aktuálním dokumentu.

Aktuálně je HTML ve verzi pět, která byla vydána v roce 2014 a od předchozí verze HTML4 z roku 1997 přináší podstatné změny. Mezi ty nejpodstatnější patří přímá podpora přehrávání multimédií. Nové, zkrácené a rychlejší zápisy značek. Autoři dávají důraz na jednoduchost a zároveň účinnost. Další novinkou oproti předchůdci je možnost vytvořit aplikaci, která funguje v prohlížeči i tehdy, když uživatel nemá k dispozici internetové připojení. Takováto aplikace ukládá data do lokálního úložiště na uživatelově počítači. Jakmile je internetové připojení opět k dispozici, může aplikace synchronizovat data se vzdáleným serverem. [3]

3.1.2 CSS

Kaskádové styly, známé také pod zkratkou CSS (z anglického Cascading Style Sheets) jsou moderním jazykem, umožňujícím účinné formátování vzhledu stránek, napsaných v jazycích HTML, XHTML či XML. Jak už slovo kaskádové napovídá, jednotlivá pravidla kaskádových stylů se mohou vzájemně překrývat, což zvyšuje jejich efektivnost. [4]

Při použití kaskádových stylů správným způsobem, dojde k naprostému oddělení vzhledu dokumentu od jeho obsahu. Toto oddělení obou vrstev (prezentační a strukturální) zvyšuje přístupnost webu a právě v něm spočívá hlavní rozdíl proti formátování s pomocí atributů, jež se používalo dříve.

CSS funguje na principu vlastnost: hodnota. Jednotlivé prvky, kterým chceme nastavit požadované vlastnosti, označujeme pomocí názvu atributu, třídy nebo id. [5]

3.1.3 JavaScript

JavaScript je multiplatformní, objektově orientovaný skriptovací jazyk, jehož autorem je Brendan Eich z tehdejší společnosti Netscape. Dnes se používá, převáže u webových technologií. Univerzální jádro jazyka je součástí webových prohlížečů a je rozšířeno o objekty reprezentující okno prohlížeče a jeho obsah. [6] JavaScript umožňuje vložit do webových stránek spustitelný kód, který umožňuje vytvářet dynamické stránky. Ty mohou obsahovat nejrůznější programy, které komunikují s uživatelem, řídí prohlížeč, či dynamicky vytváří obsah HTML. Při spouštění kódu není třeba komunikovat se serverem. Veškerou práci skriptu zajišťuje sám prohlížeč. [7]

3.1.4 Ajax

Zkratka AJAX pochází z anglického *Asynchronous JavaScript and* XML. Tato technologie umožňuje nahrávat a ukládat data na server bez nutnosti znovu načítat celou stránku. To výrazně zvyšuje uživatelsky komfort. Tato technologie se velmi často používá ve spojení s různými chaty nebo komentáři.

AJAX ve skutečnosti není žádnou novou technologií, pouze novou kombinací technologií již dávno známých, tj. HTML (nebo XHTML), JavaScriptu, XML a XMLHttpRequest. [8]

3.2 IIS server

Internet Information Services je webový server vyvinutý společností Microsoft. Obsahuje velké množství modulů, ale jeho hlavní činností je umožnit vývoj a publikování stránek na internetu. V základním nastavení podporuje veškeré webové technologie od firmy Microsoft, jako je například ASp.NET, C#.NET, C# winforms a podobné. Rovněž podporuje jazyk PHP. Podporované protokoly jsou HTTP, HTTPS, FTP, FTPS, SMTP a NNTP. [9]

Jde o třetí nejpoužívanější server po serveru Apache a Nginx. IIS server je součástí produktu Windows Server a všech edicí Windows od verze XP a vyšších. [10]

3.3 Programovací jazyk C#

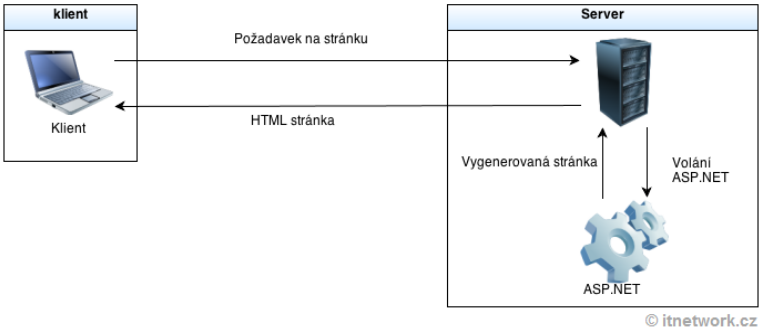
Jazyk C# je objektově orientovaný programovací jazyk, který vytvořila počítačová firma Microsoft, zároveň s platformou NET. První zmínky o něm pronikly na veřejnost v roce 2000 jako součást .NET frameworku. Jede o zjednodušenou, vylepšenou a čistě objektovou verzi programovacího jazyka C++. C# se využívá hlavně k tvorbě webových aplikací, webových služeb, univerzálních Windows aplikací a formulářových aplikací pro desktopové počítače. [11]

3.4 .NET framework

Jedná se o sadu knihoven, které umožňují tvorbu webových aplikací v programovacím jazyce C#. Knihovny ulehčují práci programátora, protože obsahují již vytvořené řešení mnoha základních problémů, které u webových technologií nastávají. Tyto problémy mohou být například: autentifikace uživatelů, zabezpečení aplikace, práce s databází, správa souborů a mnohé další. [12]

3.4.1 Princip fungování

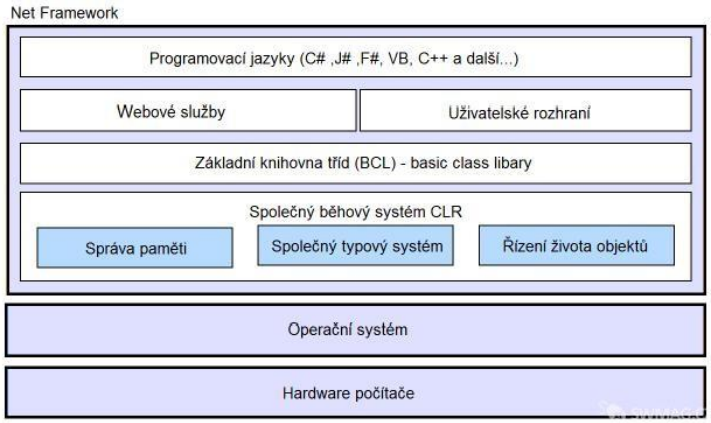
ASP.NET funguje na bázi klient-server. Klient pošle dotaz na server, který podle tohoto dotazu vygeneruje příslušnou HTML stránku a tu pošle zpět klientovi. To znamená, že klient neví, co se děje na straně serveru a zajímá ho pouze daná stránka, kterou obdrží od serveru. [13]



Obrázek 1 - Princip fungování .NET
Zdroj: https://www.itnetwork.cz/csharp/asp-net/tutorial-uvod-do-asp-dot-net

3.4.2 Struktura

Struktura je tvořena třemi základními vrstvami. První, známější vrstva se skládá z tzv. Common Language Runtime (pozn. Programovací jazyky) a Basic Class Library. Do prvně zmiňované části spadají programovací jazyky jako například C#, J#, F#, VB, C++ apod. Tato část slouží k běhu systému a skládá se z webových služeb a uživatelského rozhraní. V druhé části frameworku najdeme společný běhový systém CLR, ve kterém se dále nachází správa paměti, společný typový systém a řízení života objektů. Dalším ze základních vrstev je operační systém a v neposlední řadě se zde také nachází hardware počítače. [14]



Obrázek 2 - Struktura .NET jazyka
Zdroj: http://www.swmag.cz/670/architektura-net-framework-rozebrana-do-podrobna/

3.4.2.1 Comon Language Runtime

Pod tímto pojmem si lze představit virtuální stroj, ve kterém jsou umístěny funkce nutné k běhu této platformy a programovací jazyky zde mají k dispozici všechny knihovny

tříd systému, se kterými pracují. Tento systém obsahuje obdobu strojového kódu, který se nazývá Microsoft Intermediate Language.

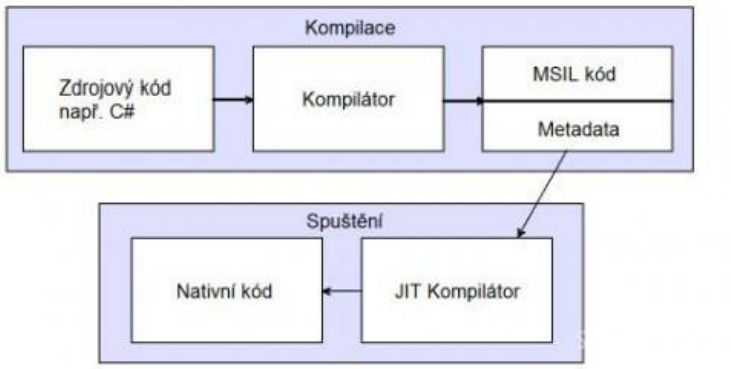
Zdrojový kód není kompilován přímo do strojového kódu, ale do MSIL kódu, což má za následek omezení výkonnosti prostředí. Tento nedostatek je vynahrazován mnohými výhodami, které MSIL přináší. Je to například řízený kód, jednotný typový systém či správa paměti.

Největší výhodou této platformy je možnost vývoje v mnoha programovacích jazycích. Tato vlastnost se nazývá Cross-language interoperability, což znamená možnost spolupráce mezi jednotlivými jazyky. [15]

3.4.2.2 Microsoft Intermediate Language

Jazyk MSIL je velice podobný assembleru. Byl vytvořen tvůrci programovacích jazyků za účelem vyvíjení nových jazyků pro platformu .NET. Při kompilaci vezme překladač kód v původním jazyce a přeloží ho do jazyka MSIL. Poté nám CLR zajistí, že při spuštění aplikace MSIL kód přeloží pomocí Just in time kompileru do strojového kódu.

Kompilace probíhá tak, že na začátku máme zdrojový kód napsaný v jakýmkoliv programovacím jazyce. Ten se následně zkompiluje do MSIL kódu, a nakonec pomocí JIT kompiléru vytvoří nativní kód. [15]



Obrázek 3 - Průběh kompilace .NET
Zdroj: http://www.swmag.cz/670/architektura-net-framework-rozebrana-do-podrobna/

3.4.2.3 Just in time Kompilátor

JIT slouží k dynamickému překladu z MSIL kódu do nativního. Zajišťuje vylepšení provozní výkonnosti programu. Rozlišujeme dva typy kompilací: interpretovanou a statickou. Interpretovaný kód je přeložen z vysokoúrovňového jazyka do strojového kódu průběžně, při každém spuštění. Statická kompilace vyžaduje tento překlad pouze jednou při prvním spuštění.

JIT navazuje na dvě myšlenky v runtime prostředí. Na binární a dynamickou kompilaci. Několik moderních runtime prostředí, jako je například Microsoft .NET Framework a také většina implementací Javy, spoléhají na JIT kompilaci pro vysokorychlostní spuštění kódu. [15]

3.5 MVC

Jedná se o architekturu, která říká, jakým způsobem bude strukturovaná daná aplikace. Aplikace je rozdělená na tři samostatné vrstvy. Na datovou, logickou a uživatelskou. Název MVC znamená model, view, controller, kde se každá z těchto komponent stará o jednu ze zmíněných vrstev. Model se stará o datovou, view o uživatelskou a controller o logickou vrstvu.

Jednotlivé vrstvy navzájem spolupracují. Požaduje-li uživatel nějakou operaci, zavolá se nejdříve logická vrstva, která si vyžádá data od datové vrstvy. Tyto data zpracuje a předá je uživatelské vrstvě, která je interpretuje uživateli.

Toto rozdělení do tří vrstev přináší značné výhody. Takto navržená aplikace je mnohem přehlednější a lépe se udržuje. Pokud je později potřeba provést změnu v jedné z vrstev, ostatní mohou být beze změny zachovány. [16]

3.6 Bootstrap framework

Jedná se o sadu nástrojů pro tvorbu webu a webových aplikací. Vyvinul ho Mark Otto a and Jacob Thornton. V roce 2011 byl publikován jako opensource projekt a v roce 2014 se stal nejsledovanějším projektem na Githubu. Framework obsahuje předem vytvořené prvky, které lze jednoduše umístit na web. To ulehčuje práci vývojářům, kteří nemusí znovu vytvářet běžné prvky, bez kterých se stránka neobejde. Nalezneme zde navigační menu, dropdown menu, tlačítka, tabulky, vyskakovací okna, formuláře, šablony stránek a mnohé další. Používání Bootstrapu je velmi jednoduché, k jeho použití stačí pouze základní znalosti HTML. [17]

3.7 Entity Framework

Entity Framework Je sada technologií v ADO.NET, které podporují vývoj aplikací orientovaných na data softwaru. Umožňuje nám objektově-relační mapování. To zajistí, že databázové tabulky se přímo mapují na C# třídy. V kódu tedy pracujeme jen s objekty a

framework sám na pozadí generuje SQL dotazy, pro práci s databází. S jazykem SQL vůbec nepřijdeme do styku a naše aplikace je celá objektově orientovaná. [18]

3.7.1 Modely pro práci s daty

S Enity frameworkem je možné pracovat dvěma způsoby. První způsob funguje tak, že se nejprve vytvoří C# třída a podle jejího obsahu se vygenerují tabulky v databázi. Tomuto modelu se říká „Code First“.

Druhý způsob předpokládá existenci již vytvořené databáze, ze které nám Entity Framework automaticky vygeneruje C# třídy, se kterými můžeme pracovat jako s běžnými objekty. [19]

3.8 CKeditor

Jedná se takzvaný „WYSIWYG“ editor. Tento název vychází z anglické věty „What you see is what you get“, česky „co vidíš to dostaneš“. Jak už je z názvu patrné, výsledný dokument bude vypadat přesně tak, jak je v editoru vytvořen. Práce s editorem je stejná jako s běžnými editory typu Word, Liber Office, Google docs a podobné.

Editor automaticky vloží do dokumentu takové HTML tagy, aby výsledná stránka odpovídala vytvořenému obsahu z editoru. Výhodou těchto editorů je možnost vytvářet dokumenty i bez znalosti HTML jazyka. Naopak nevýhoda je, že výsledný zdrojový kód může obsahovat nadbytečné či nepotřebné znaky a tím zvyšuje velikost dokumentu.

Editorů tohoto typu existuje velké množství. Já jsem si pro svou práci vybral CKeditor. Jedná se o open source projekt, jehož výhodou je jednoduchá implementace a existence online builderu, kde stačí jednoduše zvolit vzhled a funkce editoru. Toto sestavení pak stačí pouze stáhnout a přidat do projektu. [20]

3.9 Identity Framework

Jedná se o Microsoftem předpřipravené řešení, pomocí kterého lze zabezpečit aplikaci. Zajišťuje systém členství uživatelů ve skupinách, který umožňuje přidat funkci přihlášení do aplikace. Uživatelé si mohou vytvořit účet a přihlašovat se uživatelským jménem a heslem nebo mohou použít externího zprostředkovatele přihlášení jako je Facebook, Google, Microsoft Account, Twitter nebo jiné služby. Informace o uživatelích pak lze ukládat do lokálního uložiště nebo do vzdálené databáze. Dále nám umožňuje

zpřístupnit určité části aplikace jenom pro přihlášené uživatele nebo pro uživatele, kteří mají přiřazenou specifickou roli. [21]

3.10 GitHub

Jedná se o webovou službu, která umožňuje vývoj softwaru. Stará se o správu jednotlivých verzí souboru. Zaznamenává změny na jednotlivých řádcích kódu, ale také změny binárních souborů jako jsou obrázky, videa, zvukové záznamy a podobně. Poskytuje historii jednotlivých verzí, u které vždy zobrazí změny oproti aktuální verzi. Také umožňuje vytvořit jednotlivé vývojové větve. Ty je možné ukládat lokálně i do vzdáleného repositáře. Bezplatná verze poskytuje pouze veřejné vzdálené repositáře, u placené verze lze vytvořit i soukromý. [22]

4 Vlastní práce

Vlastní práce spočívá v analýze potřeb návštěvníků této webové aplikace. Na základě výsledků analýzy je vytvořen logický návrh stránek a dále se implementují potřebné funkcionality. Bude se jednat o stránky obce Hřebeč. Při vytváření stránek jsou demonstrovány klíčové vlastnosti C#.NET a použitých frameworků.

4.1 Analýza

Pomocí analýzy jsou zjištěny funkcionality, které by měly stránky obsahovat.

4.1.1 Interview s vedením obce

Vedení obce uvedlo, že obsah jednotlivých částí stránek, musí být možno upravovat a aktualizovat i bez znalosti jakéhokoliv programovacího jazyka a bez zásahů do zdrojového kódu. Správu aplikace budou zajišťovat dvě konkrétní osoby, kterým musí být umožněno se přihlásit pomocí uživatelského jména a hesla. Dále vedení vzneslo požadavek, že na stránky musí být umožněno vkládat dokumenty ve formátu PDF, Word a Excel.

4.1.2 Dotazování občanů

Dotazování občanů probíhalo jak osobními rozhovory, tak i formou elektronického dotazníku, který byl zveřejněn na facebookových stránkách obce. Z této analýzy vyplývá, že občané požadují stránku, kde budou přehledně vypsány veškeré aktuality, co se obce týkají. Do této sekce budou spadat nové vyhlášky, veškeré probíhající akce, různá ohlášení a další důležité informace podobného charakteru.

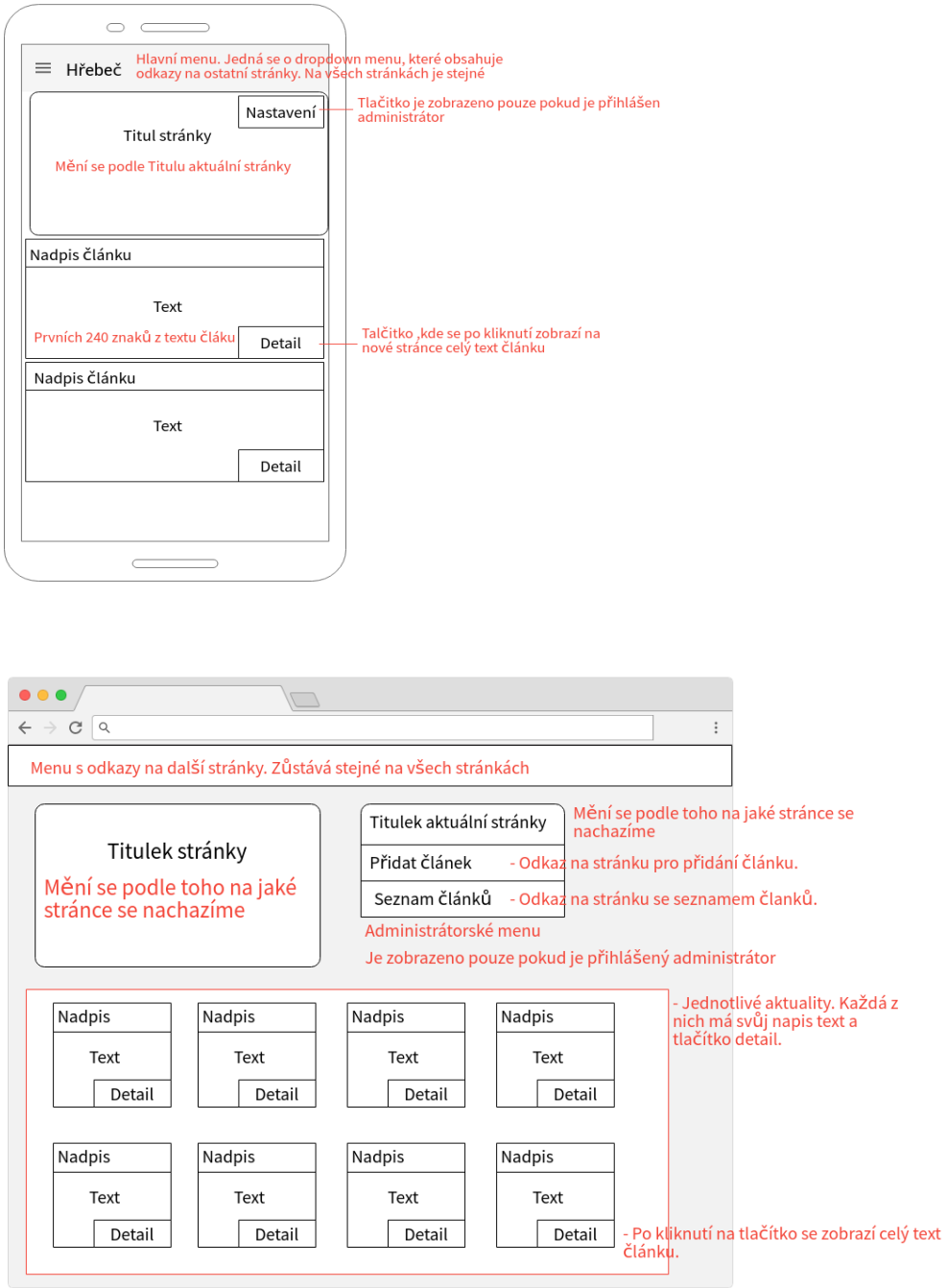
Dále vyžadují stránky, kde si budou moci vyhledat a stáhnout často používané dokumenty státní správy. Také z analýzy vyplývá, že převážná většina dotázaných přistupuje na internet pomocí svého mobilního zařízeni. Z tohoto důvodu je také důležité optimalizovat stránky pro takováto zařízení.

4.2 Logický návrh – Drátový model

Logický návrh byl vytvořen na základě požadavků vyplývajících z analýzy. K vytvoření byl použit online editor Gomockingbird. Návrhy jsou udělány jak pro mobilní zařízení, tak i pro klasické stolní počítače.

4.2.1 Domovská obrazovka

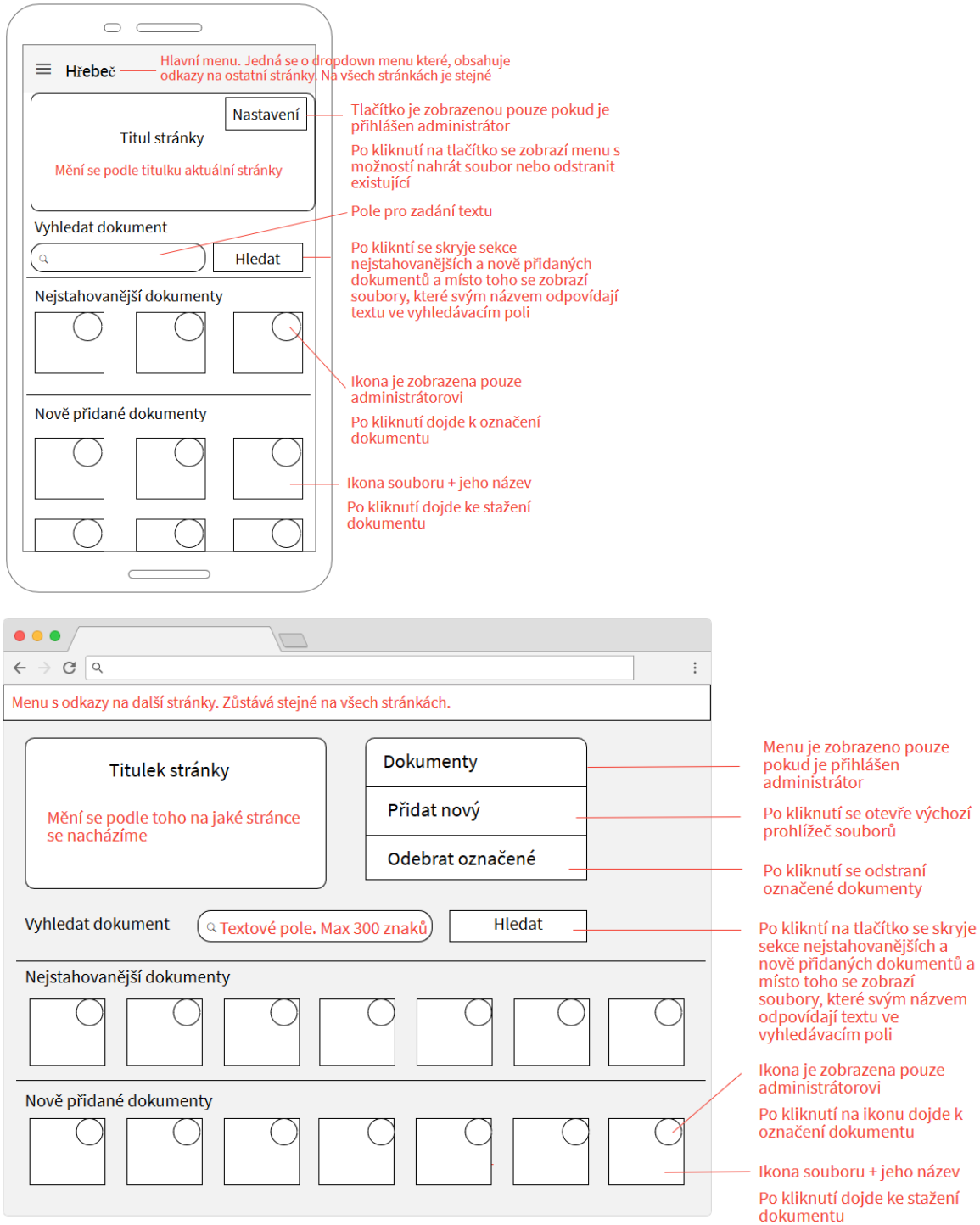
Jedná se o úvodní obrazovku, na kterou je uživatel přesměrován po zadání URL adresy. Obsahuje veškeré aktuální informace o dění v obci. Pokud je uživatel přihlášen, jako administrátor může rovněž přidávat a odebírat události.



Obrázek 4 - Drátový návrh domovská stránka
Zdroj: Autor

4.2.2 Dokumenty

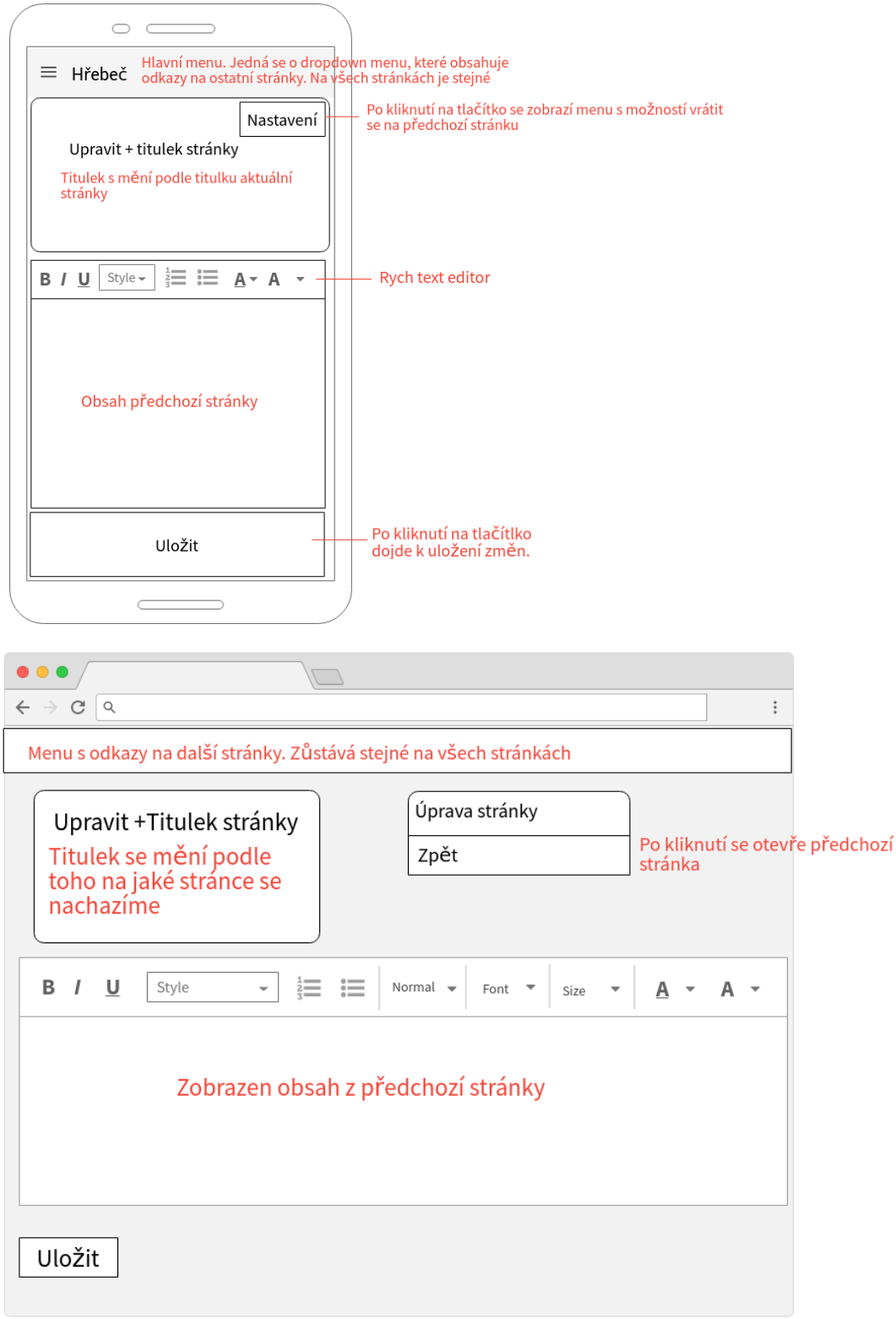
Stránka obsahuje různé dokumenty ke stažení. Jsou zobrazeny nejstahovanější a naposledy přidané dokumenty. Dokumenty lze také vyhledat podle názvu.



Obrázek 5 - Drátový návrh stránka s dokumenty
Zdroj: Autor

4.2.3 Úprava stránky

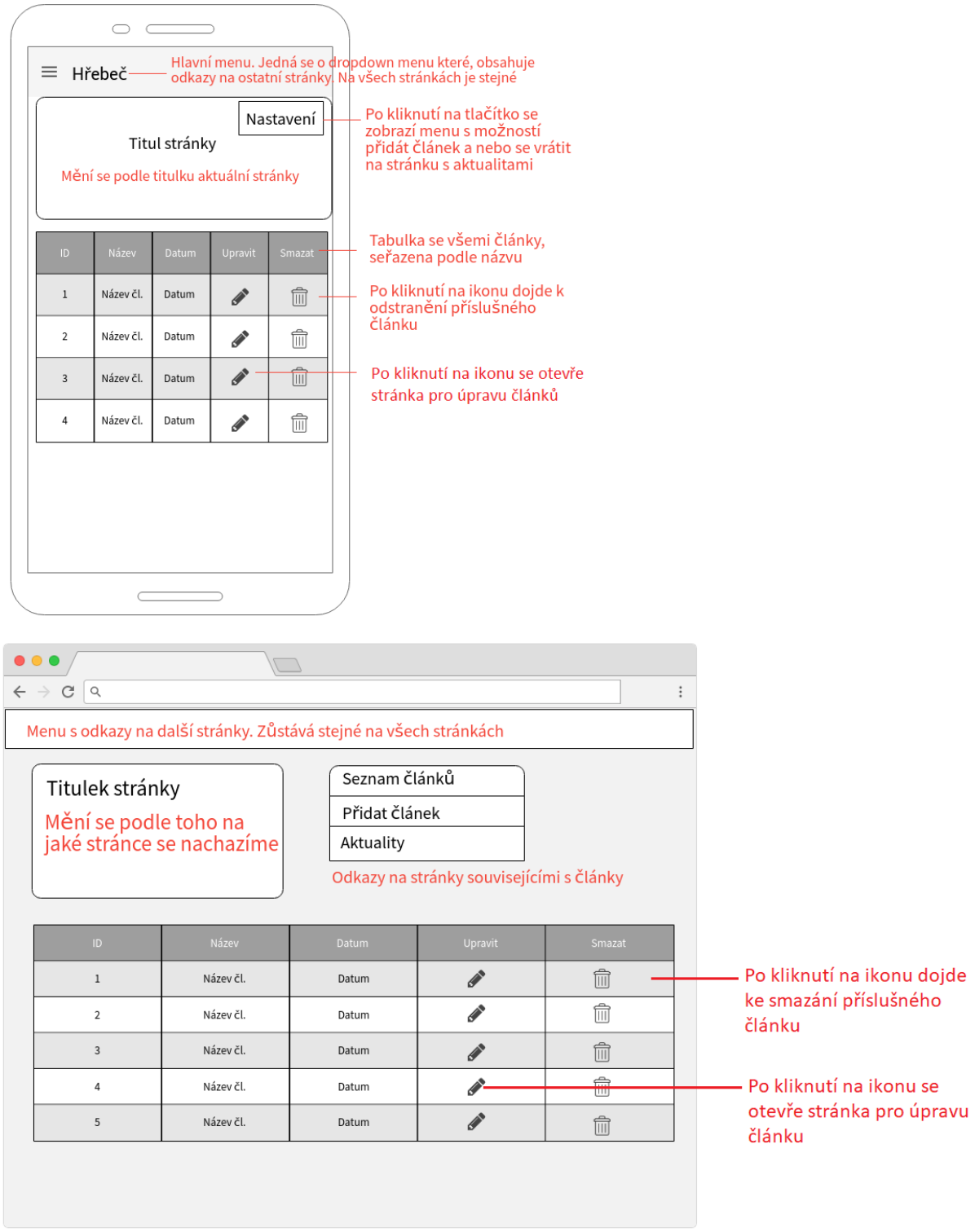
Stránka je dostupná pouze přihlášenému uživateli. Je přístupná pomocí administrátorského menu a slouží k úpravě obsahu stránky pomocí Rych text editoru.



Obrázek 6 - Drátový návrh, úprava stránky
Zdroj: Autor

4.2.4 Seznam článků

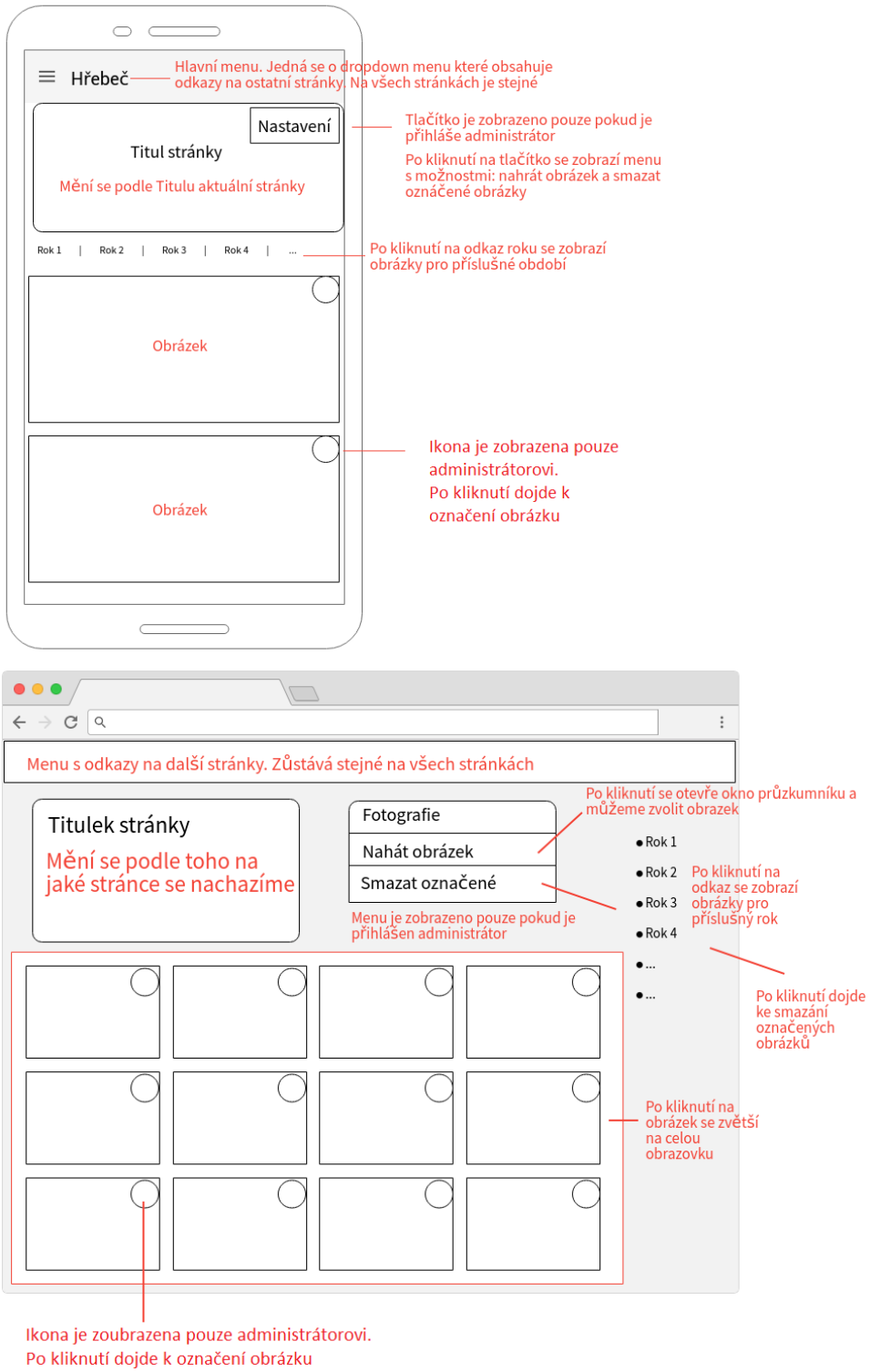
Stránka je přístupná pouze přihlášenému uživateli a poskytuje nám pohled do databáze článků, které jsou seřazeny podle názvu. Také nám umožňuje články upravit nebo odstranit z databáze.



Obrázek 7 - Drátový návrh, seznam článků
Zdroj: Autor

4.2.5 Fotografie

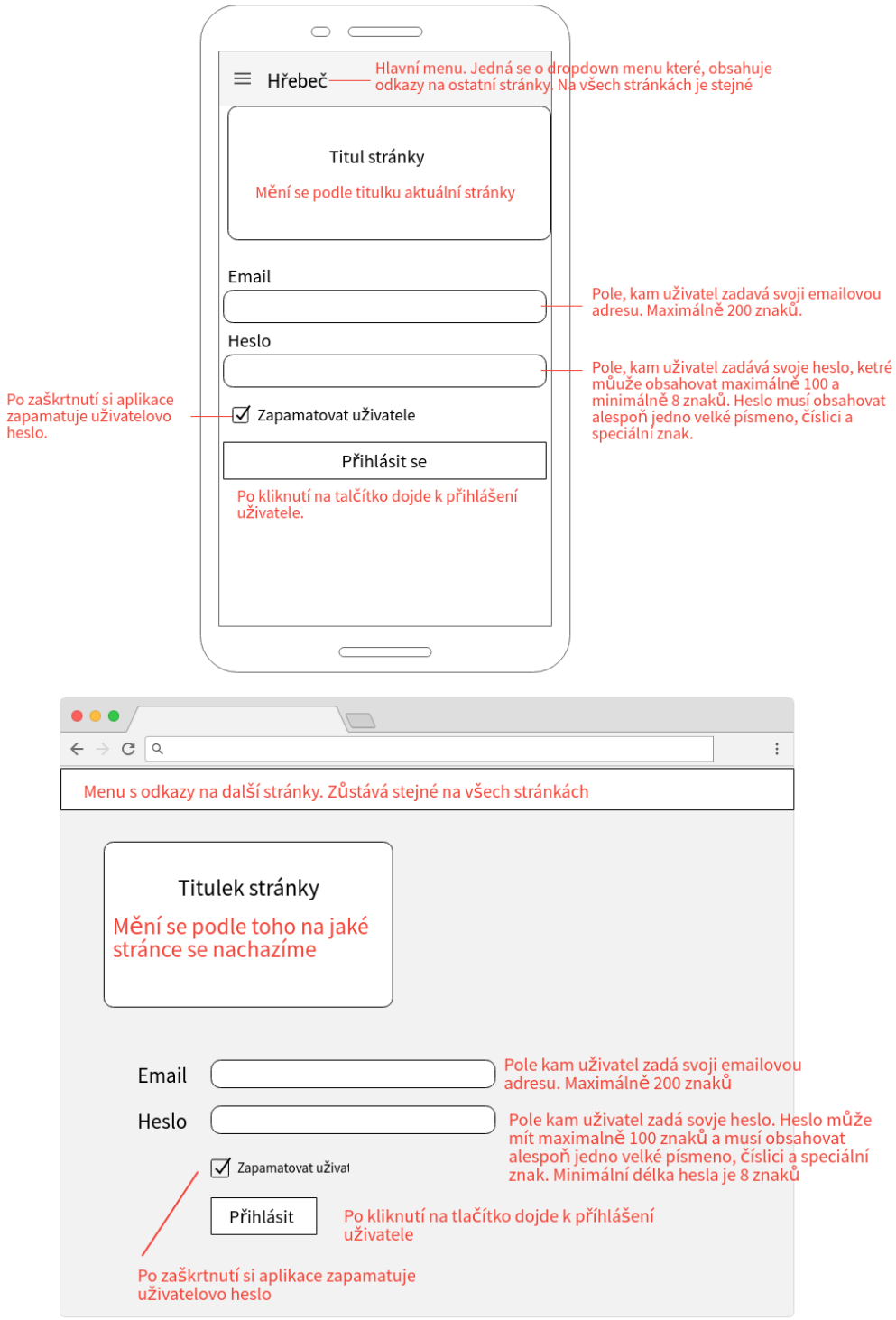
Na stránce jsou zobrazeny fotografie. Je zde možnost filtrovat fotografie podle zvoleného roku. Pokud je přihlášen administrátor může jednotlivé fotografie odebírat a přidávat.



Obrázek 8 - Drátový návrh, fotografie
Zdroj: Autor

4.2.6 Stránka přihlášení

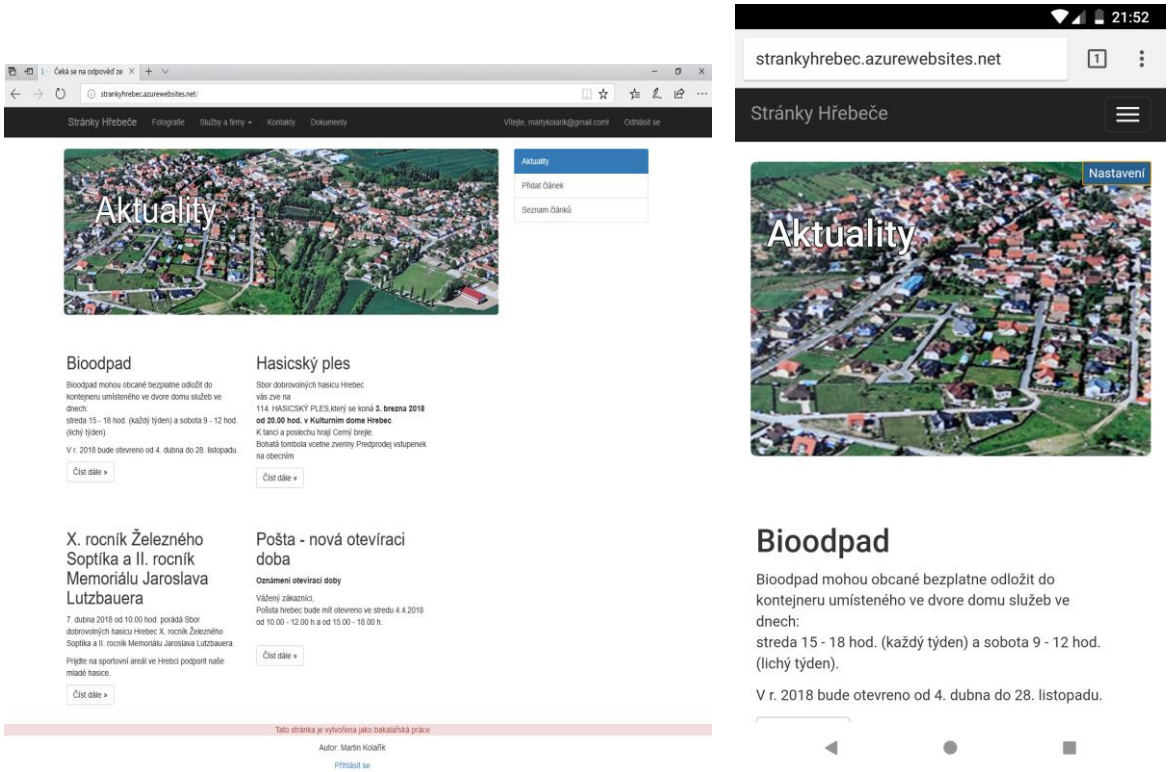
Na tuto stránku je umístěn odkaz v zápatí každé stránky. Pokud se uživatel úspěšně přihlásí, budou mu zpřístupněny veškeré administrátorské funkce.



Obrázek 9 - Drátový návrh, přihlašování
Zdroj: Autor

4.3 Grafický návrh

Z důvodu grafického sjednocení jednotlivých stánek a z důvodu co nejpřívětivějšího uživatelského rozhraní byla zvolena, jako grafický základ, šablona z Bootstrap frameworku. Její hlavní prvky jako jsou hlavní menu, nadpisy jednotlivých stránek, postranní menu, oblast pro text byly zachovány. Veškeré další prvky, které byly přidány dle logického návrhu, byly stylizovány pomocí předdefinovaných tříd, které jsou také součástí frameworku, tak aby byl zachován jednotný design.



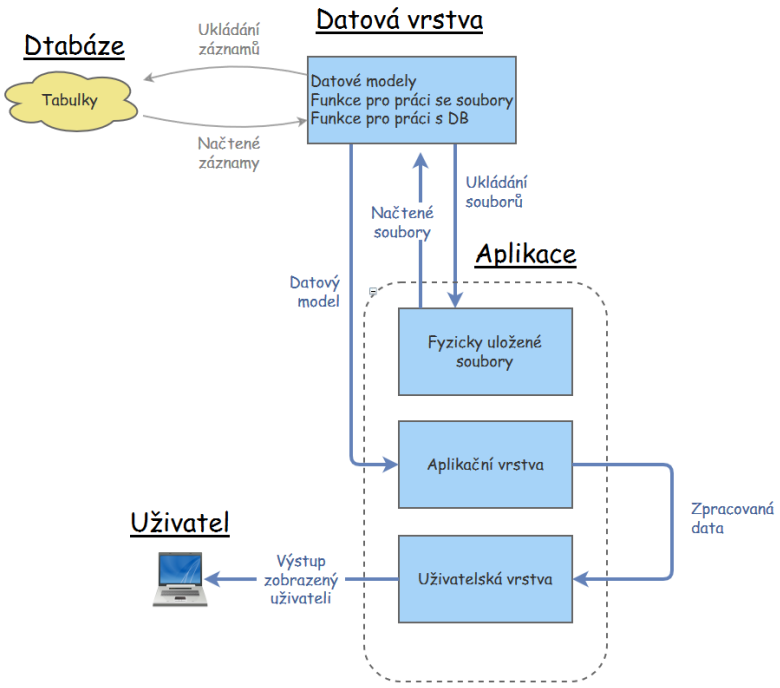
Obrázek 10 - Grafický návrh
Zdroj: Autor

4.4 Implementace

Zdrojový kód aplikace byl vytvořen pomocí programu Microsoft Visual studio. Během vývoje byl zdrojový kód spravován pomocí aplikace GitHub. Zde byly vytvořeny dvě vývojové větve. První větev byla určena pro vývoj a testování aplikace. A na druhou větev byly umísťovány stabilní a otestované verze aplikace z první větve. V každé nové stabilní verzi byly řádně okomentovány veškeré změny.

4.4.1 Struktura aplikace

Aplikace je tvořena třemi oddělenými vrstvami, které mezi sebou spolupracují a předávají si data.



Obrázek 11 - Struktura aplikace
Zdroj: Autor

4.4.1.1 Datová vrstva

Datová vrstva obsahuje veškeré datové modely a funkce, které s nimi pracují. Také jsou zde umístěny funkce pro práci s databází, které umožňují přidávat, odebírat a upravovat jednotlivé hodnoty v tabulkách databáze.

Datová část aplikace je umístěna do samostatně oddělené dll knihovny, na kterou je následně přidána reference z dané aplikace. Výhodou takto oddělené datové vrstvy je přenositelnost. Pokud bychom například chtěli k této webové aplikaci, vytvořit nějakého desktopového klienta, tak nám stačí připojit tuto dll knihovnu. Tím získáme veškeré prostředky pro práci s daty.

4.4.1.2 Aplikační vrstva

Aplikační vrstva je tvořena kontrollery, které umožňují jednotlivé funkcionality. Rovněž jsou do aplikační vrstvy ukládána data, která přímo souvisí s aplikací, jako je například obsah jednotlivých stránek nebo jednotlivé fotografie.

4.4.1.3 Uživatelská vrstva

Uživatelskou vrstvu tvoří jednotlivé pohledy, jejichž vzhled je upravován pomocí kaskádových stylů a JavaScriptu.

4.4.2 Společné zobrazení

Pokud jsou některé části stránky neměnné a opakují se na více místech aplikace, je žádoucí si vytvořit takzvanou šablonu neboli layout, která tyto části automaticky vygeneruje v každé části aplikace. Šablona musí obsahovat značku `@RenderBody()`, místo které se vygeneruje obsah požadovaného pohledu. Funguje to tedy tak, že spuštěná aplikace poskytne obsah nějakého pohledu a tento obsah se obalí kódem šablony.

Aby aplikace měla informaci o tom, že má použít šablonu, musí se ve složce s pohledy vytvořit nový pohled se jménem „_ViewStart“, ve kterém je uvedeno umístění šablony, kterou má aplikace použít. `@{Layout = "~/Views/Shared/_Layout.cshtml";}`

4.4.3 Uživatelské účty

V aplikaci musí být umožněno přihlášení uživatele a tím se identifikovat jako správce. Z analýzy vyplynulo, že stránky budou spravovat pouze dva konkrétní lidé, proto není nutné implementovat jednotlivé uživatele a jejich role. Stačí pouze ověřit, zda je daná osoba přihlášena a pokud ano, tak jsou jí zpřístupněny editační funkce.

Správa uživatelů je řešena pomocí Identity frameworku, který lze přidat ihned při vytváření projektu. Je možné si vybrat mezi přihlašováním pomocí místních účtů, pracovních účtu nebo pomocí Windows uživatele.

Při implementaci bylo zjištěno, že framework ukládá do databáze uživatelovo ID jako string hodnotu, což je nevhodné pro další práci s uživateli. Proto bylo nutné projít všechny vygenerované třídy a změnit tuto hodnotu na int a poté provést migraci, čímž se vytvořila nová databáze s již správnou hodnotou.

Po úspěšné implementaci lze velmi jednoduše a rychle měnit míru zabezpečení a přístup do jednotlivých částí aplikace. Například je možné definovat pravidla pro zadávání hesla. Ve třídě Config stačí určit požadavky, jak má heslo vypadat.

```
services.Configure<IdentityOptions>(options =>
    {
        options.Password.RequireDigit = true;
        options.Password.RequiredLength = 8;
        options.Password.RequireNonAlphanumeric = false;
        options.Password.RequireUppercase = true;
        options.Password.RequireLowercase = false;
        options.Password.RequiredUniqueChars = 6;
       });
```

Přístup do jednotlivých částí aplikace může být řízen dvěma způsoby. Pomocí prvního je přidána v kontroleru značka `[Authorize]` nad příslušnou akci. Pokud takovouto akci bude chtít spustit nepřihlášený uživatel aplikace mu zamítne přistup. Pravým opakem je pak značka `[AllowAnonymous]`, která povolí přístup všem uživatelům.

Druhou možností je omezit přístup v pohledu. Pomocí podmínky `@if(Request.IsAuthenticated)` zajistíme, že kód uvedený v podmínce bude vygenerován pouze uživateli, který je přihlášen. Tohoto využíváme například při zobrazování administrátorského menu.

```
@if (Request.IsAuthenticated)
      {
       @RenderSection("adminMenu", false)
      }
```

Spolu s touto podmínkou je vhodné používat takzvané částečné zobrazení. To nám umožnuje vložit obsah zvoleného pohledu na místo, které je určené příkazem `@Html.Partial("Partial_View")`.

4.4.4 Úprava stránky

Pro úpravu obsahu na stránce byl použit CKEditor. Jeho implementace je velmi jednoduchá. Stačí pouze stáhnout soubor na stránkách výrobce a překopírovat ho do aplikace. Zde se pak jednoduchým Javascriptem určí textové pole, místo kterého se zobrazí okno editoru.

```
    <script
type="text/javascript">CKEDITOR.replace("CKE");</script>
```

Následným testováním bylo zjištěno, že IIS server automaticky vyhodnotil práci s textem, který obsahuje HTML značky za potenciálně nebezpečný a zamítnul přístup k těmto stránkám. Z tohoto důvodu je nutné k atributu v dané třídě, která pracuje

s takovýmto textem, přidat značku `[AllowHtml]` a také k akcím v kontroléru přidat značku `[ValidateInput(false)]`.

Přidání textu stránky do pole editoru je zajištěno pomocí kontroléru, který využitím služeb datové vrstvy načte příslušný textový soubor (ve kterém je uložen obsah dané stránky) ze serveru. Poté je tento text předán pohledu, který obsahuje formulář a v něm umístěné okno editoru. Pokud je tento pohled typovým, což lze zajistit pomocí značky `@model Stranka`, vyplní automaticky daný text do formuláře. Tlačítko uložit pošle text do kontroleru, který se postará o jeho uložení na server.

```
@model Stranka
@using (Html.BeginForm("Page_Edit_Comit", "StaticPages", new {
name = Model.title }, FormMethod.Post))
{
    <div class="form-group">
        @Html.TextAreaFor(x => x.text, new { @class = "form-
control", id = "CKE" })
        <script
type="text/javascript">CKEDITOR.replace("CKE");</script>
    </div>

    <button type="submit" class="btn btn-default btn-
lg">Uložit</button>
}
```

4.4.5 Fotografie

Při ukládání fotografií na server je důležité omezit jejich velikost, aby nedocházelo ke zbytečnému plýtvání místem a ke zpomalení při načítání stránky. Tyto účely plní následující funkce, na jejíž vstup je poslán daný obrázek, jeho umístění a požadovaná kvalita. Číselná hodnota kvality vyjadřuje o kolik procent dojde ke snížení kvality oproti původnímu obrázku.

```
public static void SaveJpeg (string path, Image img, int
quality)
{
    if (quality<0  ||  quality>100)
        throw new ArgumentOutOfRangeException("quality must be
between 0 and 100.");
     EncoderParameter qualityParam = new
     EncoderParameter(Encoder.Quality, quality);
     ImageCodecInfo jpegCodec = GetEncoderInfo("image/jpeg");
     EncoderParameters encoderParams = new EncoderParameters(1);
     encoderParams.Param[0] = qualityParam;
     img.Save(path, jpegCodec, encoderParams);
}
```

4.4.6 Notifikace

Funkci notifikací je možné jednoduše implementovat pomocí dočasné proměnné TempData["název"]. Hodnota této proměnné je zachována pouze do prvního přečtení. V kontroleru se nastaví hodnota této proměnné vždy po určité akci. Dále stačí v pohledu ověřit, zda je proměnná s daným názvem nastavená na nějakou hodnotu nebo je rovna hodnotě null a případně vypsat její hodnotu. Zdrojový kód v pohledu vypadá takto:

```
@if (TempData["msg-succes"] != null)
    {
   <div class="alert alert-success alert-dismissible     fade
   in" role="alert">
      <button type="button" class="close" data-
      dismiss="alert" aria-label="Close">
          <span aria-hidden="true">×</span>
    </button>
           @TempData["msg-succes"]
  </div>
    }
```

V kódu byl použit element, který je součástí bootstrap frameworku a zajišťuje zobrazení notifikace v oblasti, kterou lze zrušit křížkem.

Obrázek 12 - Notifikace
Zdroj: Autor

4.4.7 Dokumenty

Jednotlivé dokumenty jsou ukládány do složky Documents. Tato složka je vytvořena přímo ve v aplikaci pomoci Visual studia.

Vyvolaní okna pro, procházení souborů je podporováno přímo v HTML pomocí tagu <input type=file />, ale rovněž si webový prohlížeč sám určí, jaký text u tohoto tlačítka zobrazí. Aby bylo možné změnit vzhled a text tohoto tlačítka, musí se aktuální tlačítko skrýt a vytvořit další nové tlačítko, kde se pomocí JavaScriptu vytvoří script, který po kliknutí na druhé tlačítko spustí akci tlačítka prvního.

```
<input id="upload" type="file" />
<a href="" id="upload_link"> Nahrát dokument</a>

<script type="text/javascript">
    $(function () {
        $("#upload_link").on('click', function (e) {
            e.preventDefault();
            $("#upload:hidden").trigger('click');
        });
    });
</script>
```

Jednotlivé soubory jsou pak fyzicky uloženy na server, pomocí funkce knihovny Systém.Io

```
File.WriteAllText(Server.MapPath(@"~/Stranky/" + file ), s.text);
```

4.4.8 Ukládání a načítání dat

Jednotlivý obsah stránek je ukládán v podobě textového souboru přímo na server. Veškeré obrázky a dokumenty jsou také ukládány přímo na server, ale zároveň s tím je přidán záznam do databáze s informací o umístění, názvu a datum přidání souboru. Do databáze jsou rovněž ukládány jednotlivé články spolu s jeho textovým obsahem.

Práce s databází je zajištěná pomocí Entity frameworku, který automaticky vygeneruje objekty, na základě zvolené tabulky v databázi. Díky tomu lze pracovat s aplikací jako plně objektovou a nemusíme vytvářet jednotlivé SQL příkazy pro práci s databází.

4.4.9 Práce s databází

Entity framework vygeneroval jednotlivé datové modely na základě zvolené tabulky. Aby bylo možné s těmito modely pracovat, je potřeba implementovat jednotlivé funkce, které zajistí komunikaci s databází. Základními funkcemi budou výběr z databáze na základě id, přidání záznamu, úprava již uloženého záznamu a odebrání záznamu. Jednotlivé funkce jsou demonstrovány na datovém modelu Články.

4.4.9.1 Načtení článku na základě id hodnoty

Vytvořením nové instance context třídy StrankyEntities vznikne připojení do databáze. Tato instance bude obsahovat jednotlivé tabulky databáze. Using direktiva zajistí, že po provedené funkci bude řádně ukončeno spojení s databází.

Nakonec vybereme tabulku a pomocí příkazu FirstOrDefault vyhledáme požadovaný záznam s daným id. Pokud takový záznam neexistuje vrátí se hodnota null.

```
    public static Clanky vyberZDb(int id)    {
            Clanky clZdb = new Clanky();
            using (StrankyEntities context = new StrankyEntities()){
                try {
                    clZdb = context.Clanky.FirstOrDefault(c => c.Id == id);
                }catch
                  {  return null; }
            }
            return clZdb;
        }
```

4.4.9.2 Přidání článku

Funkce pro přidání záznamu do databáze dostane na vstupu instanci datového modelu. Následně stačí tuto instanci uložit do databáze pomocí příkazu `Add` a potvrdit změny příkazem `SaveChanges`.

```
     public static void pridejDoDB(Clanky cl)
       {
           using (StrankyEntities context = new StrankyEntities())
           {
               context.Clanky.Add(cl);
               context.SaveChanges();
           }
              }
```

4.4.9.3 Odebrání článku

Odebrání záznamu funguje stejným způsobem jako přidávání, pouze místo `Add` je použit příkaz `Remove`.

4.4.9.4 Úprava existujícího článku

Po připojení k databázi se určí požadovaný záznam pomocí `FirstOrDefault`. Nastaví se jeho nové hodnoty a příkazem `SaveChanges` se uloží změny v databázi.

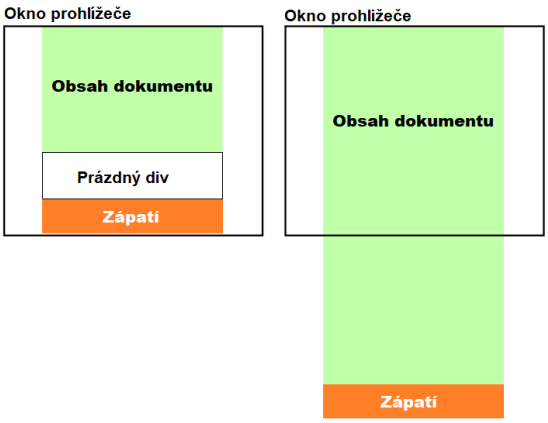
```
public static void uprav(Clanky cl)
        {
            using (StrankyEntities context = new StrankyEntities())
            {
                Clanky1 clZdb = new Clanky1();
                clZdb = context.Clanky.FirstOrDefault(c => c.Id == cl.Id);
                clZdb.Nazev = cl.Nazev;
                clZdb.Text = cl.Text;
                context.SaveChanges();
            }
        }
```

4.4.10 Úprava zápatí

Při implementaci bylo nutné vyřešit problém s umístěním zápatí. To se neumisťovalo na dolní okraj prohlížeče v případě, kdy obsah dokumentu byl kratší než zobrazovací plocha prohlížeče. Naopak pokud byl obsah dokumentu delší, než je zobrazovací plocha prohlížeče, zápatí bylo umístěno na dolním okraji okna prohlížeče, ale překrývalo se s obsahem dokumentu.

Tento problém je možné vyřešit vložením prázdného divu za končící text. Minimální výška tohoto divu je nastavena na 100 % a vzdálenost od konce dokumentu je rovna záporné výšce zápatí. Pokud bude nyní obsah delší, než je zobrazovací plocha prohlížeče, je výška divu nulová. Naopak pokud je mezi obsahem a dolním okrajem okna prohlížeče volné místo, vyplní se tímto divem.

Tím tedy zajistíme, že v případě krátkého obsahu dokumentu bude zápatí vždy na dolní hraně prohlížeče. Při dlouhém obsahu dokumentu bude umístěno ihned za tímto obsahem.



Obrázek 13 - Zápatí stránky
Zdroj: Autor

4.5 Testování

Testování probíhalo za pomocí šestice osob z různých věkových kategorií. První kategorie byla v rozmezí 15-20 let, druhá 20-45 let a poslední kategorie byla od 45let výše.

Každé osobě bylo zadáno, aby se se dostala do určitých sekcí stránky. Bylo pozorováno, kolik jim to zabere času a kde dochází k největšímu zdržení. Na základě výsledků tohoto pozorování bylo upraveno uživatelské prostředí tak, aby bylo co

nejpřívětivější. Zároveň se tím otestovalo i to, zda jednotlivé stránky plní správně svou funkcionalitu a zda jsou přístupné.

Další kritérium, které by mělo být otestováno je stabilita stránky. Simuluje se připojení velkého množství uživatelů, a sledují se ukazatele průměrné odezvy aplikace. Grafy vytíženosti webu si lze zobrazit přímo na stránkách Microsoft Azure.

4.6 Zabezpečení aplikace

Aplikace je zabezpečena pomocí uživatelských účtů, kdy uživatel musí být přihlášen, aby se dostal do určitých sekcí aplikace. Hesla účtů z hlediska bezpečnosti není vhodné ukládat do databáze v surové podobě. Aby byly dodrženy bezpečnostní standardy je nutné heslo nejprve zahashovat a až poté uložit do databáze. Samotná databáze je pak zabezpečena pomocí brány firewall. Ta zajišťuje, že k databázovému serveru se lze připojit pouze z určených IP adres.

Pokud používáme přihlašování pomocí uživatelských účtů je také vhodné použit TLS protokol, který pomocí asymetrického šifrování zabezpečuje komunikaci mezi klientem a serverem.

Uživatel by také měl mít povinnost po určitém čase změnit své heslo. To by mělo splňovat určitý formát v závislosti na míře zabezpečení, ale nemělo by být příliš složité, aby si ho uživatel dokázal zapamatovat.

V rámci bezpečnosti by měli být všichni administrátoři zaškoleni ve správném používání veškerých funkcí. Zejména špatná práce s online editorem textu může poškodit fungování stránky. Pokud by administrátor vložil pomocí online editoru do obsahu stránky nějakou HTML značku či značku scriptu mohlo by dojít k poškození struktury stránky.

4.7 Nasazení

Webová aplikace i databáze, se kterou pracuje, je publikována pomocí cloudových služeb MicrosoftAzure. Ovládat tyto služby je možné přímo z Visual studia. Stačí se zde přihlásit pomocí účtu vytvořeného na portálu MicrosoftAzure a zvolit, na kterou webovou aplikaci, chceme nahrát vytvořený kód.

4.8 Správa a údržba kódu

Během provozu stránky budou jednotliví návštěvníci pomocí formuláře dotazováni, jak jsou spokojeni s touto stránkou a co by případně změnili. Následně budou tyto podměty poslány správci stránek, který je analyzuje a vyhodnotí, zda je vhodné udělat úpravy stránky či nikoliv.

Jelikož máme aplikaci rozdělenou do tří vrstev, případné změny stačí upravit v prezentační vrstvě a ostatní mohou zůstat nezměněné.

Dále je vhodné průběžně aktualizovat použité frameworky a ostatní technologie. To lze provést pomocí správce balíčků, ve kterém jsou vypsány všechny přidané programy spolu s možností kontroly jejich aktualizací.

5 Závěr

Cílem práce bylo vytvořit webové stránky obce. Tohoto cíle bylo dosaženo za použití technologie C#.net. Klíčové vlastnosti a výhody této technologie byly podrobně popsány v kapitole 3.4. Pro vytvoření moderních, uživatelky přívětivých stránek, je nutné použit i ostatní technologie pro tvorbu webových aplikací, jako je CSS, Javascript a podobné. Jejich princip fungování a propojení s C# .net je uveden v kapitole 3.1. Dále v praktické části byla provedena analýza jednotlivých potřeb uživatelů i administrátorů spravujících tyto stránky. Na základě těchto požadavků je vytvořen logický návrh a popsané funkcionality jednotlivých prvků na stránce. Pomocí těchto návrhů a popsaných funkcionalit je programem VisualStudio napsaný program, který bude odpovídat požadavkům z předchozích kroků. Po vytvoření programu probíhala fáze testování. To bylo prováděno pomocí šestice osob. Zjištěné chyby byly opraveny a znovu otestován daný kód.

V hotové a otestované aplikaci byl vytvořen účet administrátora, který může prostřednictvím aplikace přidávat nové účty. Stránky jsou během svého životního cyklu upravovány na základě zpětné vazby od uživatelů.

Vytvořením této bakalářské práce jsem si ověřil, že v dnešní době, kdy existuje spousta frameworků a podpůrných stránek pro vytváření webové aplikace, není důvod, aby obecní webové stránky měly zastaralý grafický design a nepodporovaly responzivní zobrazení.

V prací byly použity především online zdroje. Z důvodu rychle vznikajících nových technologií a postupů, na které nejsou knihy schopny reagovat dostatečně rychle, jelikož jejich vytvoření a vydání trvá poměrně dlouhou dobu.

Vytvořené stránky jsou dostupné na adrese http://strankyhrebec.azurewebsites.net/. Celé řešení včetně všech zdrojových kódů je k dispozici na portálu Github pod adresou https://github.com/xkolm036/StrankyObce_BP .

6 Seznam použitých zdrojů

[1] *Zbynekmlcoch* [online]. MUDr. Zbyněk Mlčoch, 2012 [cit. 2018-03-11]. Dostupné z: http://www.zbynekmlcoch.cz/informace/texty/pocitace-internet/jaky-je-rozdil-server-portal-web-internetova-stranka-prohlizec

[2] CASTRO, Elizabeth a Bruce HYSLOP. *HTML5 a CSS3: názorný průvodce tvorbou WWW stránek*. Brno: Computer Press, 2012. ISBN 978-802-5137-338

[3] *Arstechnica* [online]. [cit. 2018-03-11]. Dostupné z: https://arstechnica.com/information-technology/2014/10/html5-specification-finalized-squabbling-over-who-writes-the-specs-continues/

[4] *Garth* [online]. [cit. 2018-03-07]. Dostupné z: http://www.garth.cz/uvod-do-css/zakladni-znaky-a-principy-css/

[5] *W3schools* [online]. [cit. 2018-03-11]. Dostupné z: https://www.w3schools.com/css/

[6] *Garth* [online]. [cit. 2018-03-07]. Dostupné z: http://www.garth.cz/uvod-do-javascriptu/charakteristika-javascriptu/

[7] *JavaScript* [online]. [cit. 2018-03-11]. Dostupné z: https://developer.mozilla.org/en-US/docs/Web/JavaScript/About_JavaScript

[8] *Garth* [online]. [cit. 2018-03-07]. Dostupné z: http://www.garth.cz/ajax/

[9] *Microsoft IIS server* [online]. [cit. 2018-03-07]. Dostupné z: https://www.iis.net/

[10] Používané servery. *W3Tech* [online]. [cit. 2018-03-07]. Dostupné z: https://w3techs.com/technologies/details/ws-microsoftiis/all/all

[11] *Tutorialspoint* [online]. [cit. 2018-03-07]. Dostupné z: https://www.tutorialspoint.com/csharp/csharp_overview.htm

[12] *It network* [online]. David Čápka, 2014 [cit. 2018-03-07]. Dostupné z: https://www.itnetwork.cz/csharp/asp-net/tutorial-uvod-do-asp-dot-net

[13] *Itnetwork* [online]. [cit. 2018-03-11]. Dostupné z: Itnetwork [online]. [cit. 2018-03-11]. Dostupné z: https://www.itnetwork.cz/csharp/asp-net/tutorial-uvod-do-asp-dot-net

[14] *Microsoft.com* [online]. [cit. 2018-03-11]. Dostupné z: https://docs.microsoft.com/en-us/dotnet/framework/get-started/overview

[15] *Swmag* [online]. Zbyněk Maier, 210n. l. [cit. 2018-03-07]. Dostupné z: http://www.swmag.cz/670/architektura-net-framework-rozebrana-do-podrobna/

[16] *Microsoft - .net MVC* [online]. Microsoft [cit. 2018-03-07]. Dostupné z: https://msdn.microsoft.com/en-us/library/dd381412%28v=vs.108%29.aspx?f=255&MSPPError=-2147217396

[17] *Bottstrap* [online]. Bottstrap [cit. 2018-03-07]. Dostupné z: https://getbootstrap.com/docs/4.0/about/overview/

[18] *Entity Framework Tutorial* [online]. [cit. 2018-03-07]. Dostupné z: http://www.entityframeworktutorial.net/what-is-entityframework.aspx

[19] *Enity framework* [online]. [cit. 2018-03-07]. Dostupné z: https://docs.microsoft.com/en-us/aspnet/core/data/ef-mvc/intro

[20] *CKEditor* [online]. [cit. 2018-03-07]. Dostupné z: https://docs.ckeditor.com/ckeditor4/latest/guide/dev_basics.html

[21] *Microsoft identity framework* [online]. [cit. 2018-03-07]. Dostupné z: https://docs.microsoft.com/cs-cz/aspnet/core/security/authentication/identity?tabs=visual-studio%2Caspnetcore2x

[22] *Github* [online]. github [cit. 2018-03-07]. Dostupné z: https://github.com/