Česká zemědělská univerzita v Praze

Provozně ekonomická fakulta

Katedra informačního inženýrství

Diplomová práce

Manažerský informační systém správy automobilů

Bc. Radim Mužíček

© 2020 ČZU v Praze

ČESKÁ ZEMĚDĚLSKÁ UNIVERZITA V PRAZE

Provozně ekonomická fakulta

ZADÁNÍ DIPLOMOVÉ PRÁCE

Radim Mužíček

Systémové inženýrství a informatika
Informatika

Název práce

Manažerský informační systém správy automobilů

Název anglicky

Management information system of car management

Cíle práce

Cílem diplomová práce je vytvořit modul pro správu automobilů do manažerského informačního systému. Bude provedena analýza, návrh a implementace procesů správy automobilů, jako jsou rezervace a půjčování automobilů, správa tankovacích karet, možnost převodu informací a rezervačního formuláře do formátu PDF.

Metodika

Diplomová práce se bude skládat ze dvou hlavních části – teoretické a praktické části. Zpracování teoretické části bude založeno na studiu a analýze odborných informačních zdrojů. Na základě zjištěných poznatků budou sepsána východiska pro zpracování praktické části práce.

Praktická část práce bude spočívat v analýze a návrhu procesů, návrhu modulu informačního sytému a jeho implementaci. Analýza a návrh systému budou zpracovány pomocí jazyku BPMN a UML. Implementace bude provedena s využitím relační databáze a jazyků PHP, HTML a JavaScript.

Doporučený rozsah práce

50 – 60 stran

Klíčová slova

informační systém, správa automobilů, UML, BPMN, PHP, Javascript

---

Doporučené zdroje informací

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

KLČOVÁ, H. – SODOMKA, P. *Informační systémy v podnikové praxi.* Brno: Computer Press, 2010. ISBN 978-80-251-2878-7.

PILGRIM, M. *Ponořme se do HTML5.* Praha: CZ.NIC, 2015. ISBN 978-80-905802-6-8.

SCHILDT, Herbert. Java 8: výukový kurs. Přeložil Jakub GONER. Brno: Computer Press, 2016. ISBN 978-80-251-4665-1.

SCHILDT, Herbert. Mistrovství – Java. Brno: Computer Press, 2014. Mistrovství. ISBN 978-80-251-4145-8.

Předběžný termín obhajoby

2019/20 LS – PEF

Vedoucí práce

Ing. Marek Pícka, Ph.D.

Garantující pracoviště

Katedra informačního inženýrství

Elektronicky schváleno dne 19. 2. 2020

Ing. Martin Pelikán, Ph.D.

Vedoucí katedry

Elektronicky schváleno dne 19. 2. 2020

Ing. Martin Pelikán, Ph.D.

Děkan

V Praze dne 20. 03. 2020

Čestné prohlášení

Prohlašuji, že svou diplomovou práci "Manažerský informační systém správy automobilů" jsem vypracoval(a) samostatně pod vedením vedoucího diplomové práce a s použitím odborné literatury a dalších informačních zdrojů, které jsou citovány v práci a uvedeny v seznamu použitých zdrojů na konci práce. Jako autor uvedené diplomové práce dále prohlašuji, že jsem v souvislosti s jejím vytvořením neporušil autorská práva třetích osob.

V Praze dne 06.04.2020 _____

Poděkování

Rád bych touto cestou poděkoval všem, kteří mi byli při tvorbě mé diplomové práci nápomoci. Zejména mojí rodině a přítelkyni za trpělivost, mému vedoucímu práce panu Ing. Marku Píckovi, Ph.D., za odborné rady a konzultace během vedení této diplomové práce.

Manažerský informační systém správy automobilů

Abstrakt

Tato diplomová práce je zaměřena na nové prvky již stávajícího informačního systému spolu se specifickými požadavky na správu automobilů. Po sepsání požadavků na informační systém probíhá analýza a specifikace včetně návrhu nového rozšířeného informačního systému. Systém bude probíhat ve zkušebním provozu z důvodu odhalení možných nedostatků. Po ukončení zkušebního provozu bude systém spuštěn v plném provozu.

V diplomové práci budou definované některé typy informačních systémů se stručným popisem každého zmiňovaného typu. Budou také definovány způsoby modelování informačního systému jako například model vodopád, model spirála, prototypový model a agilní metodika modelování. Postupně se představuje kapitola o modelovacích technikách. V této oblasti se seznamujeme s UML a CASE nástroji.

Informační systém využívá jazyky PHP a HTML s pomocí kaskádového stylu CSS a skriptovacího jazyka JavaScript s pomocí knihovny jQuery, MySQL a dalších jazyků podle potřeb.

Klíčová slova: informační systém, správa automobilů, UML, BPMN, PHP, JavaScript

Management information system of car management

Abstract

The diploma thesis focuses on the new elements of the already existing information system along with the specific requirements for the car management. After the requirements for the information system is drawn up, the analysis and specification including the suggestion for the new expansion of the information system take place. The system will be running in the field test because of the detection of possible deficiencies. When the field test is finished, the system will be started in the full operation.

In the diploma thesis will be defined some types of the information systems with a brief description of each. Also, the methods of modelling such as waterfall model, spiral model, prototyping model and agile method of modelling will be defined. The next chapter introduces the modelling techniques. In this area we are dealing with the UML and CASE tools.

The information system uses PHP and HTML languages with the help of CSS cascading style sheet and JavaScript scripting language with the help of jQuery library, MySQL and other languages if needed.

Keywords: information system, car management, UML, BPMN, PHP, JavaScript

Obsah

1 Úvod ... **13**

2 Cíl práce a metodika ... **14**

2.1 Cíl práce a metodika ... 14

2.2 Metodika ... 14

3 Teoretická část ... **16**

3.1 Použitá technologie ... 16

3.1.1 Kaskádové styly CSS ... 16

3.1.2 Programovací jazyk PHP ... 16

3.1.3 Knihovna jQuery ... 17

3.1.4 Databázový server MySQL ... 18

3.1.5 Jazyk HTML ... 19

3.1.6 JavaScript ... 20

3.1.7 HTTPS ... 20

3.2 Informační systém ... 21

3.3 Životní cyklus informačního systému ... 21

3.3.1 Zadání a specifikace požadavků ... 22

3.3.2 Analýza a specifikace systému ... 23

3.3.3 Návrh ... 23

3.3.4 Implementace ... 24

3.3.5 Testování ... 24

3.3.6 Implementace systému ... 25

3.3.6.1 Souběžná strategie ... 25

3.3.6.2 Pilotní strategie ... 25

3.3.6.3 Postupná strategie ... 25

3.3.6.4 Nárazová strategie ... 25

3.3.7 Zkušební provoz ... 26

3.3.8 Servis a rutinní provoz ... 26

3.3.9 Reengineering ... 26

3.4 Modely životních cyklů informačního systému ... 26

3.4.1 Model vodopád ... 26

3.4.2 Prototypový model ... 27

3.4.3 Model spirála ... 28

3.4.4 Agilní metody tvorby IS ... 29

3.4.4.1 Extrémní programování ... 30

3.4.4.2 Vývoj řízený testy ... 31
3.4.4.3 SCRUM ... 31
3.5 BPMN ... 33
3.6 UML diagram ... 34
3.6.1 Diagram případů užití ... 36
3.6.2 Diagram tříd ... 37
3.6.2.1 Vztahy v diagramu tříd ... 38
4 Praktická část ... 42
4.1 Informační systém ... 42
4.1.1 Požadavky ... 42
4.1.2 Analýza systému ... 44
4.1.2.1 Use Case diagram ... 44
4.1.2.2 Business Process Model and Notation ... 48
4.1.2.3 Diagram tříd ... 50
4.2 Návrh ... 52
4.2.1 Wireframe ... 52
4.3 Implementace ... 53
4.3.1 Databáze ... 53
4.3.2 Přidávání nových položek ... 54
4.3.2.1 Tankovací karty ... 55
4.3.2.2 Správa automobilů ... 58
4.3.2.3 Události ... 63
4.3.3 Testování ... 67
4.3.3.1 Přístupnost ... 68
4.3.3.2 Rychlost ... 68
4.3.3.3 Validita kódu ... 68
4.4 Zhodnocení ... 69
4.5 Výsledky a diskuse ... 70
5 Závěr ... 71
6 Seznam použitých zdrojů ... 72
7 Přílohy ... 75

Seznam obrázků

Obrázek 1 - Architektura MySQL serveru 19
Obrázek 2 - Schéma JavaScriptu 20
Obrázek 3 – HTTPS 21
Obrázek 4 – Vodopádový model 27
Obrázek 5 - Prototypový model 28
Obrázek 6 - Model spirála 29
Obrázek 7 - Extrémní programování 31
Obrázek 8 - Scrum model 32
Obrázek 9 - BPMN model 34
Obrázek 10 - Ukázka Use Case diagramu 37
Obrázek 11 - Notace třídy 38
Obrázek 12 - Vztah 1 39
Obrázek 13 - Vztah 2 39
Obrázek 14 - Vztah 3 40
Obrázek 15 - Vztah 4 40
Obrázek 16 - Use Case diagram 45
Obrázek 17 - Rezervace automobilu Obrázek 18 - Událost 49
Obrázek 19 - Diagram tříd 51
Obrázek 20 - Wireframe: seznam událostí 52
Obrázek 21 - Wireframe: událost 53
Obrázek 22 - Seznam použitých tabulek v databázi MySQL 54
Obrázek 23 – Tlačítka 55
Obrázek 24 - Seznam tankovacích karet 56
Obrázek 25 - Vzhled karty 57
Obrázek 26 - PDF u tankovací karty 58
Obrázek 27 - Správa historie automobilu 58
Obrázek 28 - Záložky u automobilu 59
Obrázek 29 - QR kód 59
Obrázek 30 - Link na kalendář 60
Obrázek 31 – Kalendář 60
Obrázek 32 – E-mail technikovi 61
Obrázek 33 - Rezervace automobilu 62
Obrázek 34 – Soubory 63
Obrázek 35 – Událost 64
Obrázek 36 - Událost 65
Obrázek 37 - Zpráva vedoucímu události 66
Obrázek 38 - Test validace HTML 69

Seznam tabulek

Tabulka 1 - Vzorový scénář.....46
Tabulka 2 - Přihlášení na událost.....46
Tabulka 3 - Nahrání souboru 47

Seznam použitých zkratek

XP – Extreme Programming (Extrémní programování)

SW – Software

HW - Hardware

IS – Informační systém

NASA – National Aeronautics and Space Adminostration (Národní úřad pro letectví a kosmonautiku)

HTTPS – Hypertext Transfer Protocol Secure

HTTP – Hypertext Transfer Protocol

SLL – Secure Sockets Layer

TLS – Transport Layer Security

PHP – Hypertext Preprocessor, původně Personal Home Page

HTML – Hypertext Markup Language

CSS – Cascading Style Sheets

DOM – Document Objekt Model

CERN – Conseil Européen pour la recherche nucléaire (Evropská organizace pro jaderný výzkum)

NCSA - National Center for Supercomputing Applications (Národní středisko USA pro superpočítačové aplikace)

IBM – International Business Machines (Americká mezinárodní technologická společnost)

RAD - Rapid Application Development, též Rapid-application Development

BPMN - Business Process Model and Notation

UML - Unified Modeling Language

OMT - Object Modeling Technique

OMSE - Object-Managed Software Engineering

RUP - Rational Unified Process

UC – Use Case

WCAG - Web Content Accessibility Guideline

W3C – The World Wide Web Consortium

1 Úvod

V současné době absolvuji praxi u nejmenované automobilové firmy. Zde mám možnost pracovat ve vývojovém oddělení společnosti. Hlavními činnostmi a odpovědnostmi skupiny v rámci vývoje jsou konektivita a smartphone integrace. Jako oddělení zajišťujeme vývoj a funkčnost v oblastí telefonie, SmartLink a vývoje mobilních aplikací včetně jejího testování a jejího uvolnění pro veřejnost. Jelikož je potřeba neustále testovat aplikace a konektivitu na testovacích vozech různých specifikací, je důsledkem toho častá nedostupnost testovacích automobilů. To platí i pro mobilní telefony.

Myšlenka rozšíření mě napadla v rámci pracovní náplně, a právě díky tomu vznikla tato diplomová práce. Zefektivněný informační systém bude mít velkou šanci uchytit se ve firemní společnosti, jelikož zde není žádný jiný systém, který by obsahoval informace o testovacích automobilech a telefonech. Tento systém by měl ulehčit práci testerům, analytikům a dalším zainteresovaným osobám při testování.

V současné době je na trhu velké množství mobilních telefonů od nejrůznějších výrobců. Tato zařízení mají v sobě nespočet funkcí a aplikací a jelikož jsou tyto aplikace odlišných specifikací, bylo z tohoto důvodu zapotřebí vymyslet a naprogramovat systém, který by nám roztřídil a zpřehlednil všechny údaje.

V práci se také zaměříme na některé způsoby modelování informačního systému jako je model vodopád, model spirála, prototypový model a agilní metoda modelování. V neposlední řadě se budeme zabývat UML.

Velkou výhodou aktualizovaného informačního systému je, že uvidíme, zda jsou tankovací karty, které budou do systému nově implementovány, zapůjčené, kdo si je zapůjčil a jaká je jejich denní útrata. Dále zde bude umožněno zobrazení informacích o kartách ve formátu PDF. Pro snadnější vyhledávání jednotlivých automobilů je vytvořen QR kód. Též u automobilů uvidíme, zda jsou zapůjčené a bude k nim možné zobrazit historii zapůjčení ve formátu PDF. Ovládání aplikace bude přehledné a jednoduché.

Cílem a hlavně výstupem této práce bude webová aplikace, vytvořená v programovacím jazyce PHP (Hypertext Preprocessor). Stránka bude kódována v programovacím jazyce HTML (HyperText Markup Language) s grafickou úpravou v CSS (kaskádových stylech). Data se budou ukládat do tabulek typu MySQL.

2 Cíl práce a metodika

2.1 Cíl práce a metodika

Cílem diplomové práce je rozšíření manažerského informačního systému a vytvoření modulu pro správu automobilů. Informační systém bude rozšířen na základě zpětné vazby kolegů, kteří mají zkušenosti z provozu současného systému.

Bude provedena analýza, návrh a implementace procesů rozšíření systému a správy automobilů, jako jsou rezervace a půjčování automobilů, vytvoření a správa tankovacích karet, možnost nahrání a stažení souborů u automobilů, vytvoření a správa událostí, možnost převodu historie zapůjčení a útrat u tankovacích karet do formátu PDF, možnost převodu historie zapůjčení automobilů ve formátu PDF a vytvoření QR kódu u automobilů. Pro lepší přehlednost o vypůjčených automobilech a jejich době nám bude také sloužit nově vzniklý kalendář.

2.2 Metodika

Diplomová práce se bude skládat ze dvou hlavních částí – teoretické a praktické části. Zpracování teoretické části bude založeno na studiu a analýze odborných informačních zdrojů.

Seznámíme se s modelem vodopád, modelem spirála, prototypovým modelem a agilní metodou modelování. Na základě zjištěných poznatků bude sestaven návrh informačního systému, který bude východiskem pro zpracování praktické části práce.

Analýza a návrh systému budou zpracovány pomocí BPMN a UML. Implementace bude provedena s využitím relační databáze a jazyků PHP, HTML, kaskádového stylu CSS, skriptovacího jazyka JavaScript, správy databáze MySQL a knihovny jQuery.

Praktická část bude na základě návrhu informačního systému spočívat v implementaci rezervace a půjčování automobilů, správy tankovacích karet a správy událostí. Ve formátu PDF zde bude umožněno zobrazování historie zapůjčení tankovacích karet a jejich útrat, ale také zobrazení historie zapůjčení automobilů. Též bude do systému zakomponován nový kalendář pro přehlednější vyhledávání informací o výpůjčkách automobilů. Systém bude také schopen vytvořit QR kód k automobilům pro jejich rychlejší a snadnější vyhledávání.

Praktická část bude zpracována pomocí grafické notace BPMN a pomocí UML. Dále jsme využily jazyků PHP, HTML, kaskádového stylu CSS, skriptovacího jazyka JavaScript, správy databáze MySQL a knihovny jQuery.

3 Teoretická část

3.1 Použitá technologie

3.1.1 Kaskádové styly CSS

Kaskádové styly slouží v informatice jako jazyk, pomocí kterého popisujeme způsob zobrazení elementů na stránkách napsaných v programovacím jazyce jako je XML, XHTML nebo HTML. Vydané verze jsou CSS1, CSS2, CSS3 a v roce 2011 byla dokončena revize verze CSS 2.1. [1]

Soubor obsahující informace o stylu je uložen v samostatném souboru s koncovkou CSS, což nám umožňuje identicky formátovat více dokumentů HTML. Aby bylo možné využívat kaskádové styly, je potřeba mít napsanou cestu k tomuto souboru v hlavičce stránky. [2]

Ukázka kódu upravující základní formátování textu v kaskádovém stylu:

```
#hlavicka {
width: 430px;
height: 350px;
margin-top: 7px;
padding-top: 11px;
background-color:#1c0eff;
vertical-align: bottom;
}
```

3.1.2 Programovací jazyk PHP

PHP je zkratka pro Hypertext Preprocessor (český překlad Hypertextový preprocesor). První verze byla představena v roce 1995. Od té doby prošel jazyk PHP mnoha vývoji od verze PHP 1.x až po současnou podobu, kterou je verze označovaná jako PHP 7.3.4. [3]

Jedná se o velmi populární a rozšířený skriptovací jazyk zásluhou své jednoduchosti, což dokazuje i podílové využití na internetu, které je přes 78 %. [4] [5]

Jazyk se velmi dobře hodí pro internetové stránky a webové aplikace. Dále se dá využít i při tvorbě konzolových her. [5]

PHP podporuje mnoho knihoven, které se dají využít například k práci se soubory, se zpracováváním textu, v grafice nebo k přístupu většině databázových souborů jako jsou MySQL nebo Oracle. Většina poskytovatelů hostingu nabízí podporu PHP jazyka zdarma. [3]

Ukázka nejzákladnějšího skriptu v PHP:

<?php echo "Ahoj, Wasah!"; ?>

3.1.3 Knihovna jQuery

Knihovna jQuery je svobodný a otevřený software pod licencí Massachusettského technologického institutu (MIT). Vymyslel ji John Resig v lednu roku 2006. Software je založen na oddělení HTML vzhledové části a struktury. JQuery je framework pro JavaScript. Jedná se o prohlížeč. Programátorovi webových stránek usnadňuje a zrychluje psaní, nejvíce u opakujících se příkazů v JavaScriptu. [6]
Můžeme se naučit z webové stránky vybírat:

- Konkrétní elementy
- Měnit jejich obsah
- Měnit styly [6]

JQuery využívá CSS selektory a mezi jeho základní funkce patří změna a procházení DOM elementů. [6]
Poskytuje:

- Tvoření efektů
- Tvoření animací
- Úpravu kaskádových stylů [6]

U jQuery považujeme za nejdůležitější vzájemné působení mezi JavaScriptem a HTML.
Fungování:

- JQuery existuje jako jeden JavaScriptový soubor obsahující všechny funkce
- Tento soubor si stáhneme a vložíme ho do zdrojového kódu dané webové stránky
- Poté ho na HTML stránce, na které ho budeme chtít využít, nalinkujeme [6]

Existují i jiné možnosti. Může být načtena pomocí Google AJAX Libraries API. Tento způsob má mnoho výhod (například unifikované cáchování a snížení odezvy). Další způsob načítání jQuery je možné přímo ze serverů Google. Tento způsob má tu výhodu, že všichni uživatelé Google mají tuto knihovnu načtenou v cache (vyrovnávací) paměti. [6]

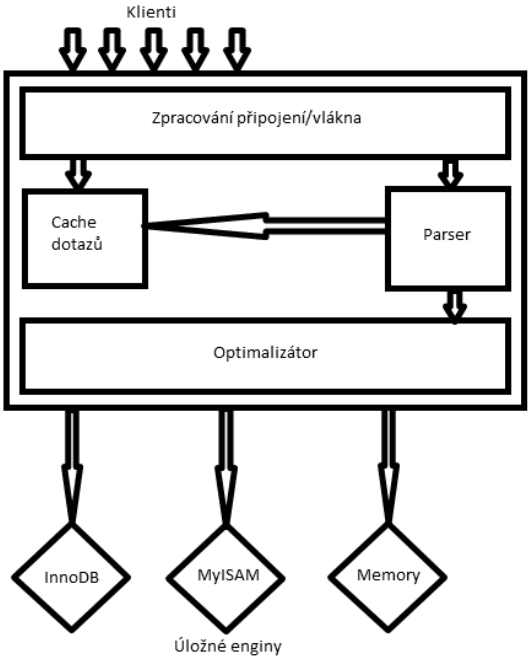
Ukázka kódu jQuery, který dělá to, že po kliknutí na text zmizí:

```
$(document).ready(function(){
$("p").click(function(){
$(this).hide();
});
});
```

3.1.4 Databázový server MySQL

Jedná se o systém řízení báze dat využívající relační databázový model. První vydaná verze byla v roce 1995 švédskou společností MySQL AB, která zanikla roku 2008. V současné době tento server vlastní společnost Oracle Corporation, která vyvíjí relační databáze. [7]

Ostatní známé databázové systémy jsou například Oracle, Informix nebo Progress. Jednou z výhod MySQL je možnost pracování s programovacími jazyky jako je C, C++, Java, PHP, Visual Basic nebo .NET. Další výhodou je bezplatné využívání SW. Nejčastěji se využívá v kombinaci s programovacím jazykem PHP, u kterého nám umožňuje přístup k uloženým datům. Naopak nevýhodou je málo možností na rozdíl od velkých databázových systémů. [8]



Obrázek 1 - Architektura MySQL serveru
Zdroj: [cs.wikipedia.org/wiki/MySQL]

3.1.5 Jazyk HTML

Hypertext Markup Language (zkráceně HTML) patří v informatice do kategorie značkovacích jazyků pro tvorbu webových stránek, což znamená, že se jedná o obohacování textu o dodatečné informace. Jedná se o hlavní programovací jazyk při tvorbě stránek v systému World Wide Web, jenž nám umožňuje zveřejňovat dokumenty na internetu. HTML je nástupcem univerzálního značkovacího jazyka Standard Generalized Markup Language. Vývoj jazyka byl velmi ovlivněn webovými prohlížeči. [9]

V roce 1989 pro CERN, což je Evropské centrum jaderného výzkumu, nacházející se poblíž města Ženeva, pracovali Tim Berners-Lee a Robert Cailliau na propojeném informačním systému. [9]

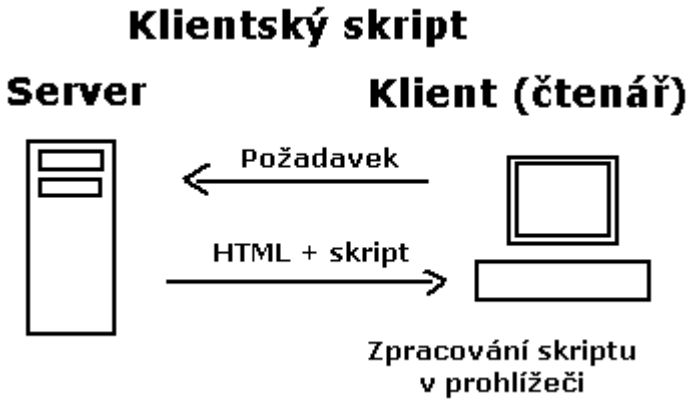
Jazyk HTML vznikl roku 1990 a s ním současně i protokol HTTP (HyperText Transfer Protocol – protokol pro přenos hypertextu). Společně s tímto vývojem vytvořil Tim Berners-Lee první webový prohlížeč (anglicky World Wide Web). [9]

O rok později CERN zprovoznil svůj vlastní web. Ve stejném roce středisko NCSA (National Center for Supercomputing Applications) vyzvala Marca Andreessena a Erica Binu k tvorbě prohlížeče Mosaic. Prohlížeč vznikl o dva roky později,

tedy v roce 1993, ve verzi určené pro počítače od společnosti IBM. Jednalo se o první prohlížeč s grafickým uživatelským rozhraním, který slavil velký úspěch. [10]

3.1.6 JavaScript

Autorem objektově orientovaného, multiplatformního a událostmi řízeného skriptovacího jazyka je Brendan Eich, narozený v roce 1961 v městě Pittsburgh. JavaScript patří z pohledu syntaxe do skupiny jazyků jako C, C++ a Java, ale z pohledu sémantiky se jedná o jiný programovací jazyk. Jedná se o klientský skript, který se většinou aktivuje až po načtení stránky na internetu. [11] [12] [13]



Obrázek 2 - Schéma JavaScriptu
Zdroj: [https://www.jakpsatweb.cz/javascript/javascript-uvod.html]

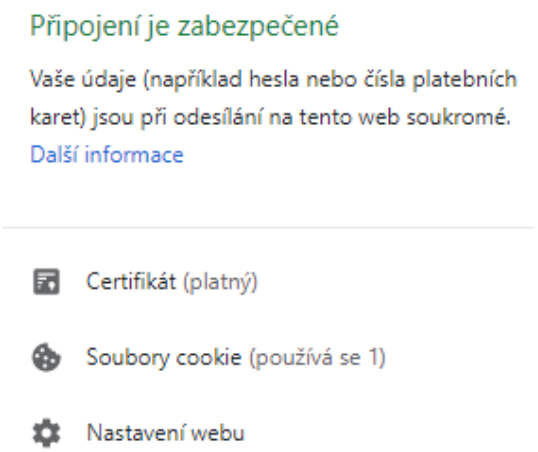
JavaScript nám umožňuje využívat funkcionality jako na příklad zvuky, interaktivní obsah nebo dynamické nabídky. [12] [13]

Nesmíme se však nechat zmást tím, že JavaScript obsahuje slovo Java. V tomto případě se jedná pouze o marketingový tah. Jediné, co mají společného, je podoba syntaxe. Programovací jazyk Java je samostatná skupina. [12] [13]

3.1.7 HTTPS

HyperText Transfer Protocol Secure (zkráceně HTTPS) je protokol, který v informatice dovoluje zabezpečenou komunikaci v síti mezi webovým prohlížečem a serverem. V roce 1994 byl vyvinut společností Netscape Communications, sídlící ve Spojených státech amerických a byl primárně určen pro webový prohlížeč nesoucí název Netscape Navigator. HTTPS má jako základ protokol HTTP, který je kombinací buď protokolu TLS nebo SLL. Jednou z výhod je ověření identity, integrita obsahu nebo třeba

důvěryhodnost dat. Na druhou stranu tu jsou nevýhody jako pokles výkonu u starších hardwarů a mírnější složitost konfigurace webového serveru. [14]



Obrázek 3 – HTTPS

3.2 Informační systém

Informační systém je celek tvořený technickými prostředky jako je počítačový hardware, se sdruženými softwary a daty. Součástí tohoto celku jsou i lidé, kteří daný software a hardware využívají. V dnešní době existuje velké množství různých typů informačních systémů. [15]

Obecné rozdělení informačních systémů:

- Veřejný IS
  - Uchovávají data, k nímž má přístup veřejnost nebo komunita
  - Je financován například z reklam, institucí nebo z dobročinných příspěvků
- Podnikový IS
  - Je využíván čistě podnikem nebo organizací
- Přístup mají pouze pracovníci s rozdílným oprávněním [15]

3.3 Životní cyklus informačního systému

Zodpovědnou a velmi náročnou prací je vedení vývoje nového systému. Životní cyklus je tvořen životními fázemi, které jsou popsány od narození do smrti (začátkem, kdy se rozhodneme pro nový IS, až do doby, kdy jej již nepoužíváme). Životní cyklus IS

obsahuje jednotlivé etapy, pomocí kterých bychom měli dojít k určenému cíli projektu. V první řadě je potřeba zjistit, zda se investice do tvorby IS vyplatí. [15] [16]
Počet životních fází je různý. Můžeme je rozdělit do čtyř fází:

- Plánování (specification)
- Návrh (design)
- Zavádění (implementation)
- Provoz a údržba (operation and maintenance) [15] [16]

3.3.1 Zadání a specifikace požadavků

Na začátku celkového návrhu, vývoje i jakýchkoliv úprav daného systému jsou požadavky uživatelů a cíle organizace. Požadavky musíme shromáždit, v základních rysech rozebrat, spočítat dobu realizace a náklady. Je zapotřebí sestavit bodový rámec požadavků, funkcí a cílů. [15] [16]
Celkový rámcový projekt obsahuje:

- Časový plán projektu
- Zdroje nutné k řešení (finance, personál, SW, HW)
- Odhad funkčnosti, rozsah systému, ekonomické efektivnosti a návratnosti investice [15] [16]

Nástroje pro vytvoření specifikačního projektu:

- Analýza současného stavu, jejímž cílem je zjistit současný stav, nedostatky a navrhnout změny
- Zajištění seznamu požadavků uživatelů a zjištění vstupních a výstupních informací
- Zjištění problémů, které jsou známy, popis jejich důsledků a nastínění řešení [15] [16]

Cílovým dokumentem této části je dokument, který určuje:

- Účel systému
- Specifikuje uživatele a jeho základní požadavky
- Definuje části systému
- Navrhuje řešení
- Obsahuje seznamy událostí a odhady datové základy
- Technické a softwarové zajištění [15] [16]

3.3.2 Analýza a specifikace systému

Část, která je svou důležitostí klíčová. Je rozborem části předchozí. Všechny chyby ve struktuře dat i systému, které se zde nepodaří odhalit, jsou později velmi obtížně odstranitelné. To může dojít až do takové fáze, že při odstranění jedné chyby se mohou objevit další dvě. Je to důsledkem toho, že informační systém je sofistikovaný organismus, kde při pozdější fázi odhalení jedné chyby, může způsobit řetězovou reakci následujících chyb. [16]

3.3.3 Návrh

Návrh je výsledkem analýzy systému. Je potřeba vypracovat dokument, který bude podkladem pro obsah smlouvy s externí firmou o návrhu a realizaci IS, časovém harmonogramu, ceně projektu, konkrétní implementaci systému (patří sem logický datový model a fyzický datový model), podmínkách zavádění v organizaci, záručním servisu a podmínkách celkového předání IS. [16]

Prvky, které by studie měla obsahovat:

- Základní údaje o sestavitelích systému
- O externí firmě a její specifikaci
- Jde-li o systém složený z několika podsystémů, také informace o jejich dodavatelích
- Základní údaje o organizaci, pro kterou je vyvíjen. Je nutné uvést týmy zaměstnanců, kteří budou spolupracovat s externí firmou
- Podrobný popis nynějšího stavu organizace
- Globální návrh IS (logický datový model), který je plánem funkcí a dat systému bez ohledu na technologické prostředí
- Detailní návrh IS (fyzický datový model), který obsahuje funkční analýzu systému, datovou analýzu, popis veškerých datových toků v organizaci a popis funkcí řízených událostmi. Celkovým výstupem je navržení funkcí a dat systému, které jsou dány na základě prostředí, ve kterém implementujeme systém.
- Podrobný popis nainstalování IS v praxi, SW a HW studie související s nasazením nového IS

- Podrobný popis testovacího provozu systému a dále poskytování záručního servisu
- Celkový harmonogram spolupráce, do něhož patří časový harmonogram dodávky, platby, celková cena, podmínky dodání, ceny pozáručního servisu aj. [16]

Veškeré údaje musí být uvedeny v detailním provedení a v podobě, která je srozumitelná všem členům vedení, kteří se rozhodují o přijetí (tím je myšlen logický datový model). Celá studie je vypracována jako poslední dokument, se kterým se vedení setká před jejich rozhodnutím o realizaci systému. Jestliže dojde k dohodě mezi firmou a autory systému, slouží tato studie jako podklad realizace a pro podmínky předání a testování. [16]

3.3.4 Implementace

V této části se zaměříme na vlastní programování. Zde bude vybrán analytik a experti v programování, kteří jsou zodpovědní za správnost řešení. Jako podklady slouží informace, které jsme již vytvořili v předchozí části, a fyzický návrh systému. [16]

Postup práce:

- Z návrhu se definují vstupy a výstupy jednotlivých operací
- Určí se způsob jejich modifikace
- Naprogramují se veškeré funkce
- Vzájemné propojení
- Jednotlivé realizované funkce se ověří a připraví se testovací data [16]

3.3.5 Testování

Jedná se o empirickou a technickou studii kvality. Je zapotřebí otestovat veškeré možné reakce systému na zadávaná data. Pokud se během testování objeví nějaké nedostatky, je nutné je opravit. Testování se provádí mimo reálné prostředí z důvodu bezpečnosti. Příkladem jsou systémy ve zdravotnictví, letectví, jaderném průmyslu a podobně. [16]

3.3.6 Implementace systému

Do provozu firmy se zavede nový informační systém pomocí instalace. Uživatele nového systému je třeba proškolit, tudíž absolvovat školení, popřípadě dodat manuály k individuálnímu proškolení. Při školení je nejvhodnější v první řadě školit vedoucí pracovníky a následně zaměstnance v provozu.
V žádném případě se nesmí tato část podcenit, neboť by mohla u budoucích uživatelů vzniknout averze vůči novému systému a důsledkem by byl neúspěch.
Zavedení systému můžeme provést různými způsoby. [16]

3.3.6.1 Souběžná strategie

Stávající provoz je nepřerušen a současně s ním běží provoz nového systému. Provoz obou systémů trvá do té doby, dokud nebude nový systém pracovat spolehlivě. Tato strategie je bezpečná, ale velice náročná. Zaměstnanci musí vše provádět dvakrát. Na toto období se většinou najímají externí pracovníci. [16]

3.3.6.2 Pilotní strategie

Je založena na zavedení nového systému jen v určité části podniku. V určité části se provádí pilotní systém, na kterém je možné zjistit co nejvíce problémových oblastí. [16]

3.3.6.3 Postupná strategie

Používá se u velice složitých systémů, kde bývají složité vnitřní vazby. Na začátku se zavádějí primární části IS, na kterých ostatní části závisí. Po jejich ověření se postupně zavádí ostatní části až po úplné zavedení systému. [16]

3.3.6.4 Nárazová strategie

Nárazová strategie spočívá v odstranění původního systému. Plně se zavede nový systém. Tato strategie je riskantní, jelikož nový systém může spadnout a firma bude bez systému, ale šetří se u ní čas i pracovní síly. [16]

3.3.7 Zkušební provoz

Zkušební provoz je zprovoznění projektu. Poskytovatel zajistí provoz, servis a v případě potřeby odstraní chyby, či dořeší doplňující požadavky uživatelů v rámci původního návrhu. [16]

3.3.8 Servis a rutinní provoz

Servis a rutinní provoz je konečnou částí implementace systému, ve které je systém provozován a používán. Jedná se o údržbu systému (zajištění správného provozu) a úpravu parametrů aplikací. Vše musí splňovat nové požadavky uživatelů. [16]

K základním povinnostem zajištění provozu patří:

- Organizace prací na počítačích a v síti
- Zajištění přístupových práv k jednotlivým aplikacím
- Sledování činnosti počítačů a síťových prostředků
- Zajištění optimálního provozu systému
- Zabezpečení systému
- Ochrana dat před neoprávněným přístupem
- Minimalizace škod v důsledku výpadku systému
- Školení uživatelů [16]

3.3.9 Reengineering

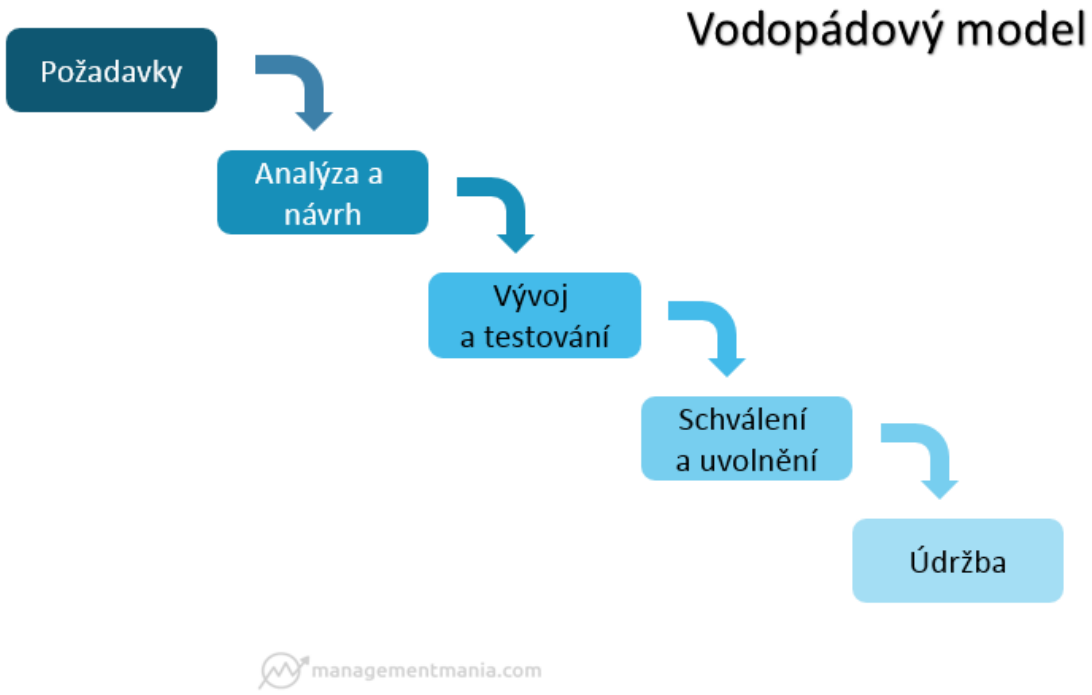
V této části přehodnocujeme požadavky na systém. Je zapotřebí zjistit, zda jsou požadavky adekvátní a reálné. Pokud je nelze upravit, pak jsme nuceni jít opět na začátek a začít znovu. [15] [16]

3.4 Modely životních cyklů informačního systému

3.4.1 Model vodopád

První formální popis vodopádového modelu je z roku 1970 od počítačového odborníka Winstona W. Royce, i když konkrétně slovo „vodopádový“ nepoužil ve svém článku. [17] [18] [19]

Jedná se o sekvenční vývojový proces, což znamená, že na vývoj je nahlíženo jako na neustále se svažující tok od fáze požadavků přes návrh, implementaci, verifikaci až k údržbě. Z toho vyplývá, že se nejdříve provede jedna fáze a až je vše z této fáze kompletní a připravené, tak se přejde k další fázi. [17] [18] [19]



Obrázek 4 – Vodopádový model
Zdroj: [https://managementmania.com/cs/vodopadovy-model-waterfall-model]

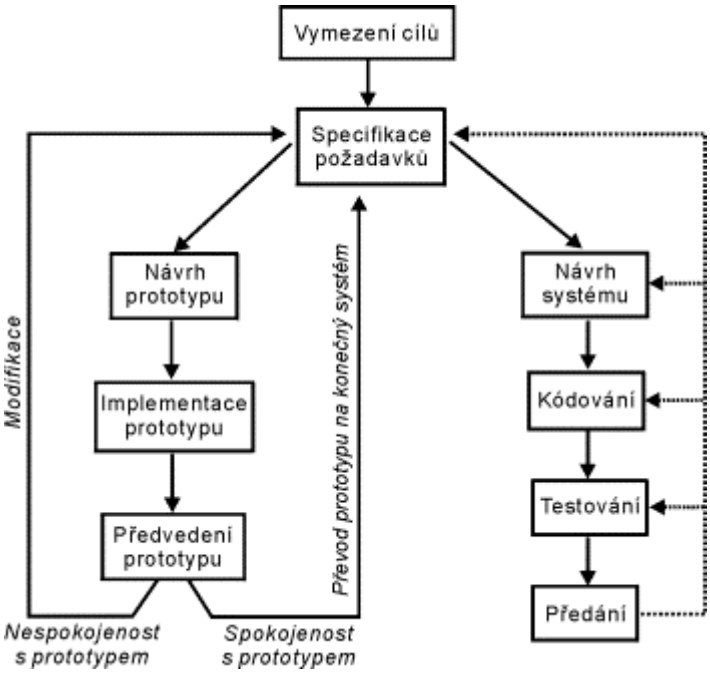
Důraz je kladen jak na dokumentaci, tak i na zdrojový kód, aby se nestalo, že v případě odchodu člena týmu by byl nástupce neschopný porozumět práci, kterou převzal, jelikož každý programátor má svůj jedinečný styl. V neprospěch hovoří fakt, že u netriviálního projektu není možné dovést jednu fázi k dokonalosti předtím, než se bude moci přejít k následující fázi.

Jedná se o široce využívaný model gigantickými vývojářskými společnostmi, mezi které patří například NASA nebo Ministerstvo obrany Spojených států. [17] [18] [19]

3.4.2 Prototypový model

Vývoj pomocí vytváření prototypů je přístup, kde je vývoj neúplných verzí softwaru, tzv. prototypu. [19] [20]

Základní principy prototypového přístupu je přístup k jednotlivým částem větších a tradičních metodik vývoje softwaru (tj. přírůstková metoda, spirála nebo RAD). Pro snížení nebezpečí projektových rizik je rozdělen na menší části a dochází k zjednodušení změn v průběhu procesu vývoje. Uživatel je začleněn v celém procesu vývoje, tím se zvýší pravděpodobnost přijetí konečné implementace uživatelem. Většina prototypů bývá vyvíjena s tím, že bude vyřazena. V určitých případech je možné pokročit od prototypu k funkčnímu systému. Pro předejití vývoje je třeba porozumět základní business problematice. [19] [20]



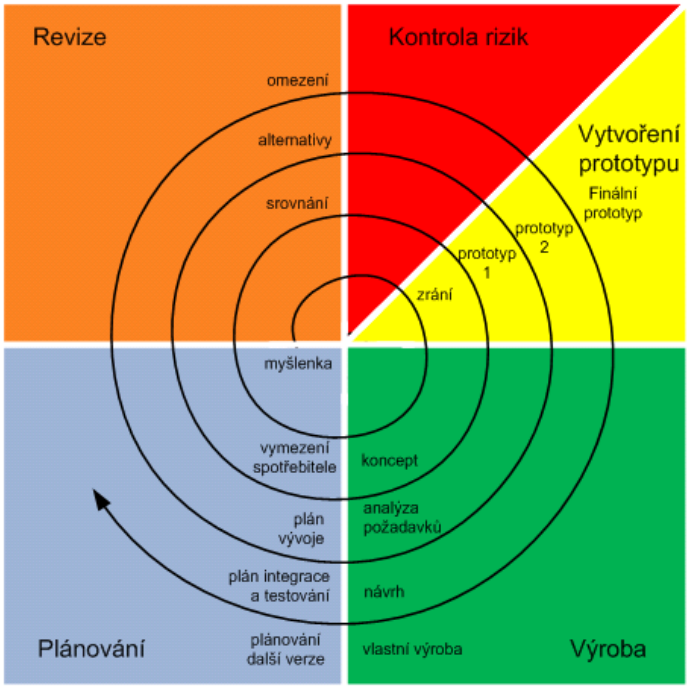
Obrázek 5 - Prototypový model
Zdroj: [fi.muni.cz/~smid/mis-zivcyk.htm]

3.4.3 Model spirála

Spirálový přístup je proces vývoje softwaru. Tento proces kombinuje prvky designového přístupu a prototypového přístupu. Kombinují se výhody obou konceptů shora-dolů (prototypování) a zdola-nahoru (designování). [19] [21]

Prvotní myšlenka spirálového přístupu se zaměřuje na analýzu rizik a minimalizaci projektových rizik, které jsou rozdělené na menší části a umožňují změny. V průběhu vývoje je možné vyhodnocovat rizika a vytvářet další možnosti projektu v průběhu cyklu. Všechny cykly spirály začínají stejným sledem kroků pro každou část produktu a pro

každou úroveň elaborace (od konceptuálních dokumentů až po programování jednotlivých programů). [19] [21]



Obrázek 6 - Model spirála

Zdroj: [umel.feec.vutbr.cz/bdts/index.php/embedded-systemy/vyvojove-modely]

V průběhu jakéhokoliv cyklu spirály jsou spouštěny čtyři základní fáze (kvadranty):

- Analýza – stanovení cílů, alternativ a rozsahu iterace
- Vyhodnocení – vyhodnocení alternativ, identifikace a řešení rizik
- Vývoj – vývoj produktu a kontrola očekávaných výsledků
- Plánování – plán pro příští iteraci [19] [21]

V úvodu každého cyklu se určují zainteresované subjekty a jejich podmínky pokládané na úspěch iterace. Na konci všech cyklů se učiní revize a předání. [19] [21]

3.4.4 Agilní metody tvorby IS

Agilní v překladu definujeme jako čilý, horlivý a aktivní. Agilní metody mají za cíl zlepšit organizaci a rozvržení práce. Tyto metody nám umožňují rychlý vývoj SW a současně umožňují reagovat na změny požadavků v průběhu cyklů. Správnost systému se ověří prostřednictvím rychlého vývoje a předložením zákazníkovi s následnou zpětnou vazbou od zákazníka. Každá agilní metoda má jiné klíčové aspekty. [22]

Jednou z častých nevýhod je ta, že zákazník nemá často ochotu denně řešit práci na projektu. Výhodou jsou zase rychlé dodávky částí projektu. [22]

Agilní metody jsou například:

- Scrum
- Extrémní programování
- Vývoj řízený testy
- aj. [22]

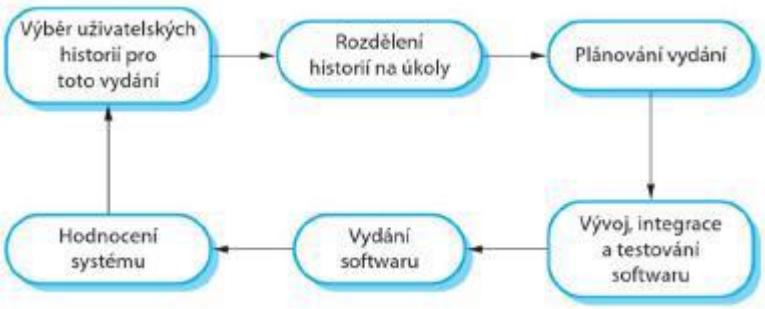
3.4.4.1 Extrémní programování

Jedná se o jednu z agilních metod vývoje SW, vytvořenou týmem okolo K. Becka počátkem 90. let. Požadavky na dosažení cíle jsou definovány jako scénáře, které jsou uskutečňovány pomocí úkolů. Jsou zde časté dodávky SW v krátkých intervalových cyklech. Využívá se zde párové programování, což znamená, že programátoři pracují ve dvojicích. [23] [24]

XP rozlišuje čtyři hodnoty:

- Komunikace
    - Je potřeba dodržovat komunikaci mezi všemi subjekty, hlavně mezi programátorem a zákazníkem, aby nedošlo k nespokojenosti s produktem ze strany zákazníka
- Jednoduchost
    - Vždy se programuje jen to, co je v daný okamžik prioritou ke splnění cílů
    - Kód by měl být jednoduchý a jasný
- Zpětná vazba
    - Výsledky testů
        - v první řadě se vytvoří test a až následně se testuje kód
        - test se skládá z jednotkových (unit) testů
    - Důležitou částí úspěchu je i zpětná vazba od zákazníka

- Odvaha
    - Další důležitá vlastnost, která je u programátora velmi potřebná. V případě, že narazí programátor na nějaký problém, musí ho opravit, avšak touto opravou může způsobit další problémy, kterých se musí vyvarovat. [23] [24]



Obrázek 7 - Extrémní programování
Zdroj: Prezentace Ing. Josef Pavlíček, Ph.D.

3.4.4.2 Vývoj řízený testy

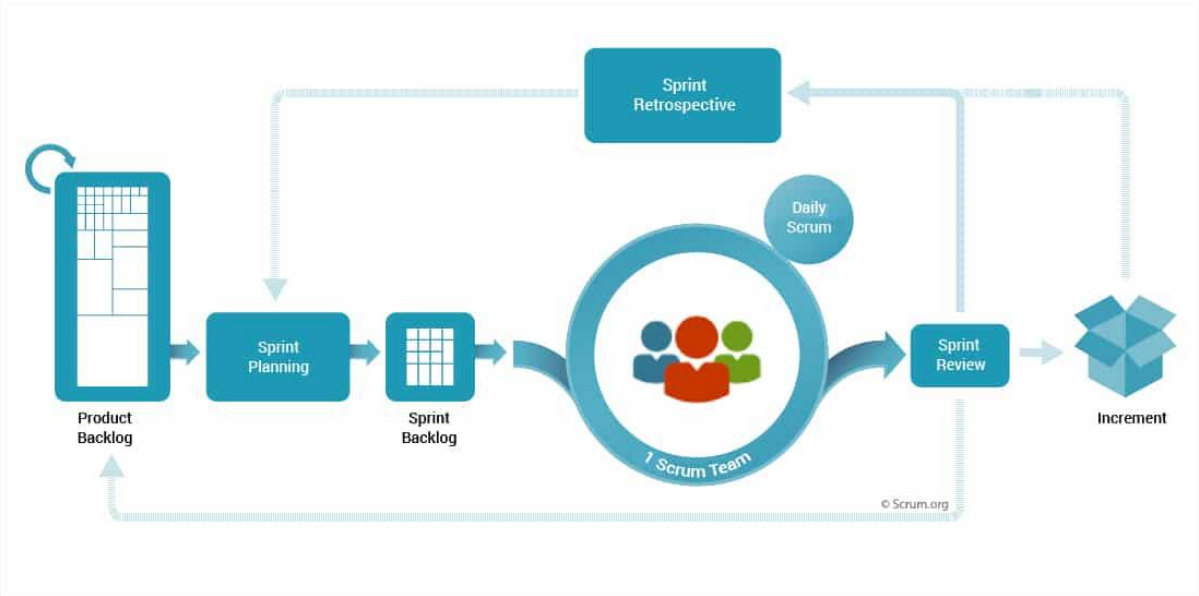
Vývoj řízený testy (též znám jako programování řízené testy) je překlad z anglického *test driven development*. Jedná se o přístup k vývoji a zakládá se na malých, ale stále se opakujících krocích, které vedou k zefektivnění celého vývoje. Nejprve je potřeba definovat si funkcionalit a následné napsání testů, které ověří funkčnost. Jako další bod je psaní kódu a na závěr úprava daného kódu. Vývoj dalších funkcionalit nepokračuje, dokud test neproběhne bez chyb. [25]

3.4.4.3 SCRUM

Jedná se o iterativní a inkrementální styl vývoje softwaru, který se řadí mezi takzvané agilní metody. Jednou z klíčových částí této metody řízení jsou každodenní krátká setkání týmu, kde každý člen informuje o své činnosti a nalezených problémech z předcházejícího dne, a dále informuje o tom, co bude dělat ten den. Tyto schůzky jsou důležité pro zjištění stavu projektu a případné implementace nových požadavků ze strany zákazníka. Z toho důvodu je kladený velký důraz na schopnosti týmu rychle dodávat části, které mají nejvyšší prioritu. [26]

Scrum se rozděluje na tři role:

- Product Owner
  - Jedná se o osobu reprezentující hlas zákazníka
  - Častá komunikace se zákazníkem
- Scrum team
  - Tým se skládá z 3 až 9 členů, kteří se zaměřují na aktuální zadanou část práce
  - Tým je multifunkční a měl by být schopen vytvořit produkt
- Scrum master
  - Osoba, která má na starosti odstraňování překážek, které brání Scrum teamu splnit stanovené cíle
  - Nejedná se o team leadera ani o projektového managera, ale spíše o osobu, která zpracovává negativní vlivy
  - Dohlíží na dodržování dohodnutých procesů ze strany členů týmu [26]



Obrázek 8 - Scrum model

Zdroj: [nutcache.com/blog/leverage-scrum-to-manage-your-projects/]

3.5 BPMN

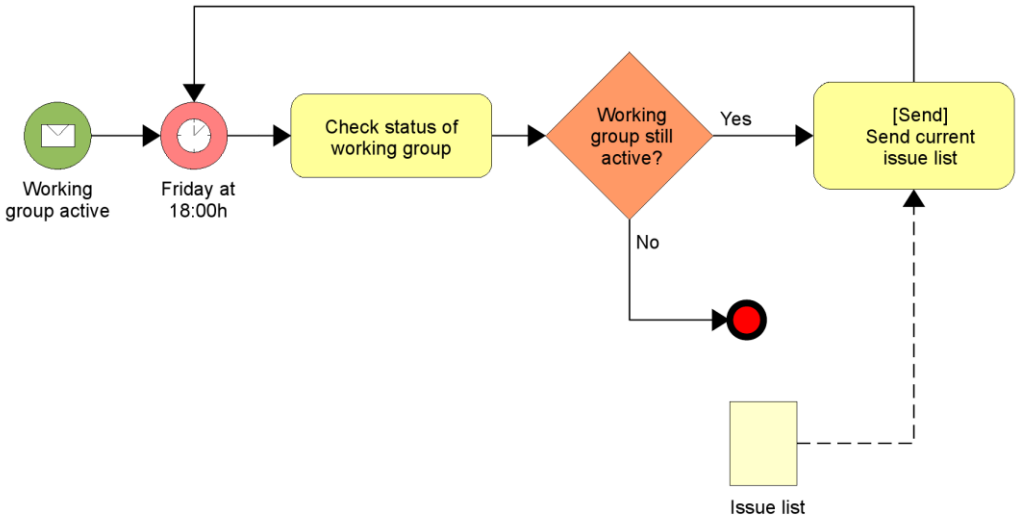
Business Process Model and Notation ve zkratce BPMN je grafická notace, která slouží k modelování podnikových procesů s využitím procesních diagramů. Současně je BPMN ratifikovaný standard Mezinárodní organizace pro normalizaci jako norma ISO/IEC 19510:2013. [27]

Od roku 2004 je BPMN vyvíjeno do současné prosazované verze 2.0.2. Dokumentace je v současné době zpracována na takové úrovni, že se vytváří předlohy, podle kterých se řeší standardní situace, se kterými se můžeme setkat během procesního řízení. To umožňuje uživateli lépe pochopit jednotlivé části BPMN diagramu.

BPMN je vytvořen k zastřešení mnoha druhů modelů a dále skrývá možnost vytvoření tzv. end-to-end podnikových procesů. [28]

Existují tři hlavní dílčí modely:

- Procesy
    - Soukromé spustitelné (vnitřní) podnikové procesy
    - Soukromé nespustitelné (vnitřní) podnikové procesy
    - Veřejné procesy
- Choreografie
- Kolaborace
    - Konverzace [28]



Obrázek 9 - BPMN model
Zdroj: [https://cs.wikipedia.org/wiki/Business_Process_Model_and_Notation]

3.6 UML diagram

UML (Unified Modeling Language) je grafický jazyk pro vizualizaci, specifikaci, navrhování a dokumentaci programových systémů. [20] [29]

UML nabízí:

- Standardní způsob zápisu návrhů systému včetně konceptuálních prvků jako jsou business procesy a systémové funkce
- Zápis konkrétních prvků jako jsou příkazy programovacího jazyka
- Databázová schémata
- Znovupoužitelné programové komponenty [20] [29]

UML podporuje:

- Přístup k analýze
- Přístup k návrhu
- Přístup k popisu programových systémů [20] [29]

UML neobsahuje:

- Způsob, jak se má používat
- Metodiku
- Jak analyzovat

- Jak specifikovat
- Jak navrhovat programové systémy [20] [29]

Vývoj UML začal v roce 1994, kdy Grady Booch a Jim Rumbaugh začali ve firmě Rational Software (nyní součást firmy IBM) spojovat své metodiky – Booch a OMT (Object Modeling Technique). [20] [29]

V září 1996 do firmy Rational Software vstoupil Ivar Jacobson se svojí metodologii OMSE (Object-Managed Software Engineering). Výsledkem jejich práce byl návrh UML (verze 0.9) a metodika RUP (Rational Unified Process). [20] [29]

Diagramy jsou nejznámější a nejpoužívanější částí standardu. Následuje přehled diagramů v UML 2.0 včetně jejich rozčlenění do skupin:

- Strukturní diagramy:
    - Diagram tříd
    - Diagram komponent
    - Domposite structure diagram
    - Diagram nasazení
    - Diagram balíčků
    - Diagram objektů, též nazýván jako diagram instancí
    - Diagram profilů
- Diagramy chování:
    - Diagram aktivit
    - Diagram užití
    - Stavový diagram
- Diagramy interakce:
    - Sekvenční diagram
    - Diagram komunikace
    - Diagram interakcí
    - Diagram časování [20] [29]

3.6.1 Diagram případů užití

Use Case Diagram (diagram případů užití) zobrazuje chování systému tak, jak ho vidí uživatel. [30] [31]

Účelem diagramu je popsat funkce systému a co od něj klient očekává. Diagram nám sděluje, co má systém umět, ale neříká, jak to bude dělat. V prvé řadě je důležité se zamyslet, co má systém umět a následně, jak to vlastně uděláme. [30] [31]

Use Case diagram se skládá:

- Z případů užití (Use Case)
- Aktérů (actors)
- Vztahů mezi nimi [30] [31]

Případ užití (zkráceně UC) je sada několika akcí, které vedou k dosažení určitého cíle. [30] [31]

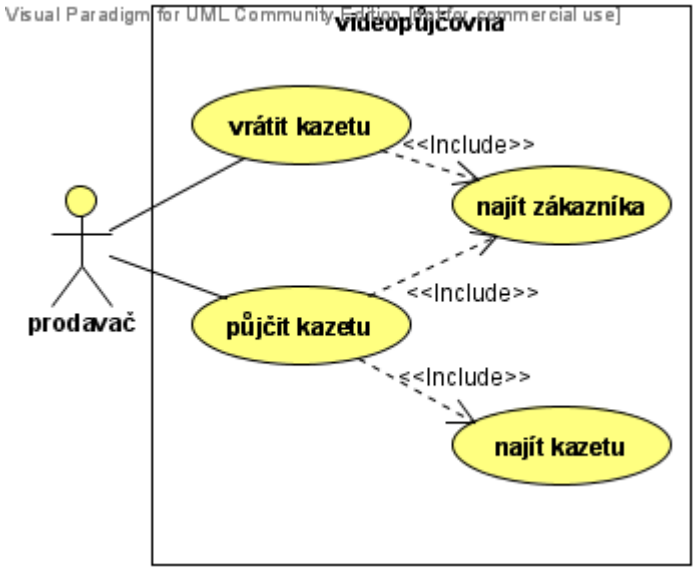
Use Case může být:

- Přidání komentáře k článku
- Registrování nového uživatele
- Vytisknutí dokumentu [30] [31]

Definuje jednu funkcionalitu, kterou by měl navrhovaný systém umět. Ta v sobě obsahuje další:

- Přidání komentáře
- Ověření uživatele
- Validaci zadaných dat
- Zápis do databáze a podobně [30] [31]

UML často hovoří o tzv. blackboxu (černé skříňce), kde skryjeme vnitřní logiku a pracujeme pouze s komponenty. Tento princip přesně využívá UC diagram. [30] [31]



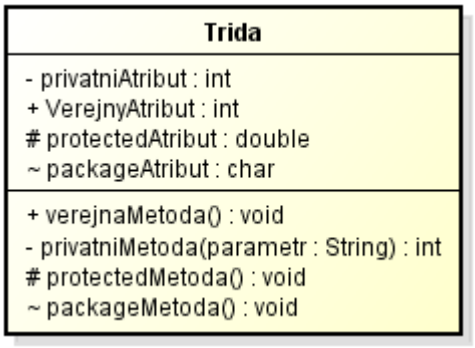
Obrázek 10 - Ukázka Use Case diagramu
Zdroj: [mpavus.wz.cz/uml/uml-b-use-case-3-2-1.php]

3.6.2 Diagram tříd

Diagram tříd (neboli Class diagram) je součástí skupiny UML diagramů. Jedná se o klíčovou část objektově orientovaného programování. Je to diagram implementace, který oproti doménovému modelu je již úplný a připravený systém, který již jako celek musí fungovat, když ho programátor přepisuje do kódu. Obsahuje všechny třídy, které informační systém bude obsahovat. Všechny třídy musí obsahovat vše, co je třeba pro jejich fungování, (tj. všechny atributy i metody). Diagram tříd je závislý na platformě, což znamená, že je specifický pro určitý programovací jazyk. V identifikátorech se již nenachází diakritika. [32] [33]

Diagram tříd je tedy jakýsi návod pro každého programátora, který mu usnadňuje práci. Díky němu je, dá se říci, programátorova práce do jisté míry mechanická, jelikož má vše již přednastavené. Programátory však učí pracovat se systémem jako s celkem, jelikož mu zamezuje napsání například jen třetiny systému. Včas mu ukáže, že mu v systému něco chybí a že daný systém nebude fungovat. [32] [33]

Složité informační systémy nelze tvořit bez tohoto návrhu, avšak je dobré si ho tvořit i pro systémy jednodušší. [32] [33]



Obrázek 11 - Notace třídy
Zdroj: [https://www.itnetwork.cz/navrh/uml/uml-class-diagram-tridni-model]

Na obrázku můžeme vidět obdélník, kde v jeho vrchní části je zobrazen název třídy, který je však, jak již bylo zmíněno, napsán bez diakritiky. Níže jsou uvedeny veškeré atributy s jejich datovými typy. Před každým atributem je uveden modifikátor přístupu. U modifikátoru přístupu máme čtyři možnosti. Je jimi + (plus), - (mínus), # (hash kříž) a ~ (tilda).

+ (plus) je znak pro veřejný atribut (public)

- (mínus) je znak pro privátní atribut (private)

(hash kříž) je znak „chráněného“ atributu (protected)

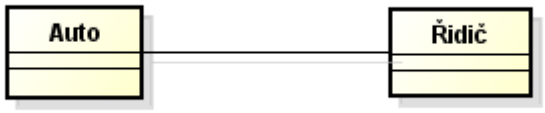
~ (tilda) je znak pro viditelný atribut v rámci balíku (package) [32] [33]

3.6.2.1 Vztahy v diagramu tříd

Vztahem v diagramu tříd nazýváme spojení mezi dvěma třídami. Je znázorněn čárou spojující dané dvě třídy. Vztah může být však také více komplexní, to znamená, že k jedné třídě se může připojit více tříd. UML nám nabízí několik druhů vztahů jako je asociace, agregace, kompozice či generalizace. U těchto vztahů je důležitá tzv. multiplicita, které se budeme věnovat níže. [34]

Asociace určuje základní vztah mezi dvěma entitami. Entity mohou existovat nezávisle na sobě. Asociaci zakreslujeme jednoduchou plnou čárou. Na obrázku č. 12 je

uveden jednoduchý vztah asociace mezi dvěma třídami (*Auto* a *Řidič*). Třída *Auto* má odkaz na druhou třídu *Řidič* a naopak třída *Řidič* má odkaz na první třídu *Auto.* [35]

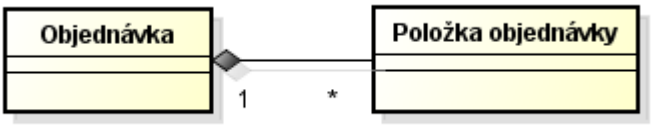


Obrázek 12 - Vztah 1
Zdroj: [https://www.itnetwork.cz/navrh/uml/uml-domenovy-model-diagram]

Dalším vztahem je agregace, která reprezentuje vztah „součást - celek“. Zakreslujeme ji jako plnou čáru zakončenou na jedné straně prázdným kosočtvercem. Ten je umístěn na straně té třídy, která reprezentuje „celek“. Tato třída drží kolekci prvků. Třída, která reprezentuje „součást“, může existovat sama o sobě a může být součástí i dalších kolekcí. Na níže uvedeném obrázku je vyobrazen příklad agregace, kde reprezentuje „celek“ třída *Sekce* a třídu „součást“ *Článek.* Na témž obrázku jsou také u vztahu uvedena čísla, která značí vztah multiplicity, která bude rozebrána níže. [35]

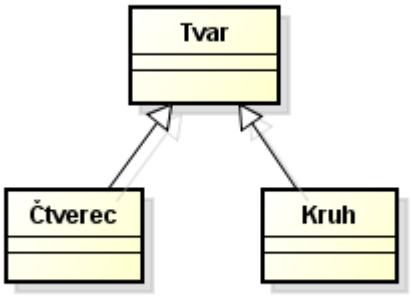
Obrázek 13 - Vztah 2
Zdroj: [https://www.itnetwork.cz/navrh/uml/uml-domenovy-model-diagram]

Vztah kompozice (též kompozitní asociace) je silnější vztah oproti agregaci, jelikož třída „celku“ nemá bez třídy „součást“ smysl. Pokud zanikne entita „celku“, zanikají tím automaticky veškeré jeho součásti. Zakreslujeme ji, též jako agregaci, plnou čárou, avšak zakončujeme ji plným kosočtvercem. U entity, která reprezentuje „celek“ musí být vždy uvedena multiplicita 1. Na obrázku č. 14 je uveden příklad tohoto vztahu u dvou tříd *Objednávka* a *Položka objednávky.* Jak již název tříd napovídá, třída „součásti“, tedy *Položka objednávky* bez třídy „celku“ *Objednávka* nedává žádný smysl. [35]



Obrázek 14 - Vztah 3
Zdroj: [https://www.itnetwork.cz/navrh/uml/uml-domenovy-model-diagram]

Vztah generalizace nám z hlediska implementace udává dědičnost. Jedna třída tedy dědí chování a vlastnosti jiné třídy. Zakreslujeme ji též plnou čarou, kterou zakončujeme na jedné straně prázdným trojúhelníkem. Trojúhelník je na straně entity, ze které jiná třída či více tříd své chování a vlastnosti dědí. Níže je uveden příklad na obrázku č. 15, kde jsou vyobrazeny tři třídy (*Tvar, Čtverec* a *Kruh).* Třídy *Čtverec* a *Kruh* dědí chování a vlastnosti třídy *Tvar*. [35]



Obrázek 15 - Vztah 4
Zdroj: [https://www.itnetwork.cz/navrh/uml/uml-domenovy-model-diagram]

Již zmíněnou multiplicitu můžeme definovat jako část vztahu, který udává, kolik instancí jedné třídy může být svázáno s instancí třídy druhé. Můžeme ji uvést u vztahů asociace, agregace i kompozice. U vztahu generalizace nelze použít, jelikož se jedná o naprosto odlišný typ vztahu (tj. o dědičnost). [35]

Možné zápisy multiplicity:

- x (číslo) – označuje konkrétní hodnotu (př. 0, 1)
- * - označuje libovolný počet (místo * se též může uvádět symbol *N*)
- 0..* (interval) – značí se pomocí dvou teček mezi dvěma hodnotami [35]

Multiplicitu můžeme značit:

- 0 - Žádná instance (značí se zcela výjimečně)

- 0..1 – Žádná nebo právě jedna instance
- 1 – Právě jedna instance
- 0..* - Žádná až více instancí
- 1..* - Jedna až více instancí [35]

Na výše uvedeném obrázku č. 13, kde jsou vyobrazeny třídy *Sekce* a *Článek*, je znázorněna multiplicita. Třída *Sekce* může mít libovolný počet *Článků* (dle uvedené * u třídy *Článek*). Třídě *Článek* patří 1 až libovolně *Sekcí* (dle uvedené 1..* u třídy *Sekce)*. [35]

Multiplicita je též uvedena na obrázku č. 14, kde jsou vyobrazeny třídy *Objednávka* a *Položka objednávky*. Třída *Objednávka* může mít libovolný počet *Položek objednávky* a třída *Položka objednávky* musí mít právě 1 *Objednávku.*

Příkladem vztahů v diagramu tříd je také obrázek č. 19 v praktické části diplomové práce.

4 Praktická část

4.1 Informační systém

Absolvuji praxi ve vývojovém oddělení u nejmenované automobilové společnosti, kde jsem se obeznámil s fungováním oddělení a jejich jednotlivými postupy. Při nastaveném postupu práce jsem od svého vedoucího dostal možnost zefektivnění již stávajícího informačního systému. Tuto možnost jsem velmi rád využil a začal na tom pracovat jako na své diplomové práci.

Současná situace je taková, že oddělení pracuje s informačním systém, který umí vyhledávat automobily ve vloženém systému automobilů. Seznam automobilů se dá filtrovat dle různých kritérií (dle druhu výbavy, motoru, paliva atd.). Také umí vyhledávat druhy mobilních telefonů z vloženého systému mobilních telefonů, které též umí filtrovat dle různých kritérií (například podle výrobce, funkcí či operačního systému aj.).

Bohužel tento informační systém nebyl pružný, a proto jsem se ho rozhodl zmodernizovat a více přizpůsobit dané situaci.

Rozhodl jsem se stávající informační systém rozšířit o modul nesoucí název *Manažerský informační systém pro správu automobilů.* Toto rozšíření přidá do systému mnoho nových inovací, které by měly ulehčit práci všem zainteresovaným osobám. Díky novému systému také dojde k pružnému a rychlému toku informací, bez nutnosti okamžitého obtěžování technika s irelevantními dotazy.

A to je další důvod, proč je implementace nového rozšíření důležitá. Zefektivní a zrychlí se práce na pracovišti, jelikož nebude třeba vyrušovat technika automobilů od jeho práce a každý bude mít možnost veškeré potřebné informace po ruce.

Informační systém jsem zrealizoval na internetové stránce www.wasah.cz. Na tuto stránku je potřeba se před vstupem zaregistrovat. Uživatel po registraci obdrží e-mail pro dokončení registrace. Po dokončení registrace je mu povolen přístup.

4.1.1 Požadavky

Jak již bylo zmíněno, stávající informační systém nebyl pružný natolik, aby se s ním nadále dalo pracovat. Proto jsem se rozhodl zohlednit všechny požadavky současných uživatelů. Obešel jsem všechny testery, analytiky a ostatní zainteresované

uživatele stávajícího informačního systému a zapsal si všechny připomínky k stávajícímu systému a všechny nápady pro zmodernizování systému.

Nové rozšíření, dle získaných informací, by mělo obsahovat:

1. Možnost rezervovat automobil na konkrétní datum a časové rozmezí přes rezervační formulář
    a. Rezervační formulář musí být možné odeslat přímo konkrétnímu technikovi, který za daný automobil zodpovídá
    b. Každý automobil by měl mít svůj vlastní rezervační formulář, aby nebylo nutné již vyplňovat informace o vozu
2. Nová kategorie tankovací karty
    a. U každé tankovací karty možnost zobrazení její historie ve formátu PDF
3. Nová kategorie událostí
    a. Každá událost bude mít určité kategorie
        - Základní informace
        - Další informace (možnost psát delší text)
        - Možnost se přihlásit
        - Možnost se odhlásit
        - Možnost nahrávání souborů k dané události
        - Možnost nahrané soubory u událostí stahovat
        - Možnost odeslání zprávy vedoucímu události
        - Zobrazení seznamu účastníků
4. Zobrazení v hlavičce aktuální čas, datum a zda je lichý či sudý týden
5. U automobilů vytvořit QR kód
6. U automobilů možnost zobrazení historie zapůjčení ve formátu PDF
7. Možnost k danému automobilu nahrávat soubory
8. Možnost nahrané soubory u automobilů stahovat
9. Individuální kalendář ke každému automobilu

4.1.2 Analýza systému

Před samostatnou realizací zmodernizovaného systému je potřeba udělat analýzu specifických požadavků. Po provedení této analýzy je nutné přejít k návrhu informačního systému, podle analýzy z požadavků.

4.1.2.1 Use Case diagram

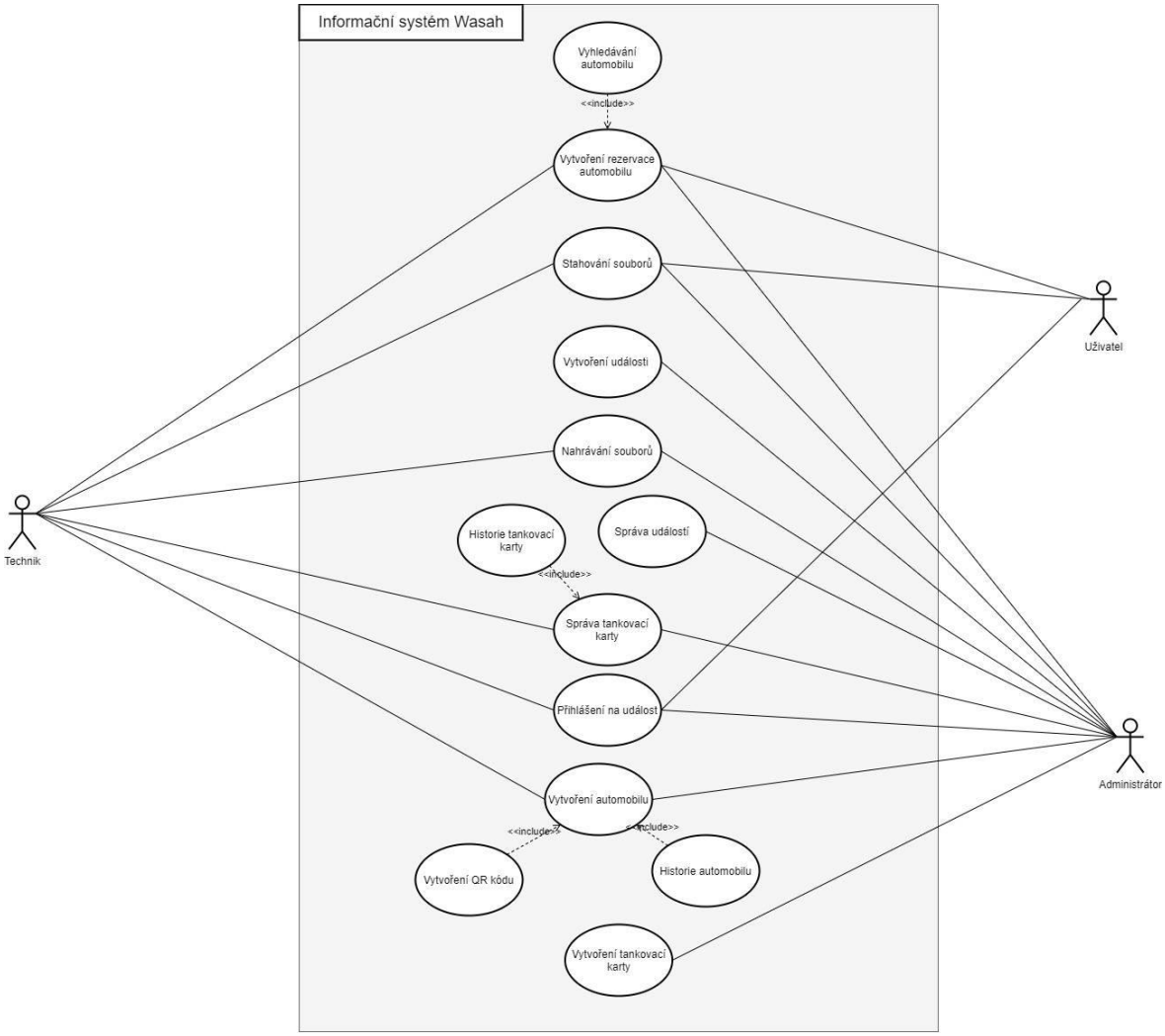
Diagram případů užití zobrazuje takové chování systému, jaké může vidět uživatel. Účelem je popsání funkcionality systému. Use case diagram spojuje všechna chování, která se vztahují k části funkčnosti systému, tj. základní normální chování a jeho variace, zvláštní a chybové podmínky, rušení a požadavky.

Use Case diagram obsahuje tři aktéry, a to *Technika, Uživatele* a *Administrátora* a definuje jejich činnosti na uvedeném obrázku č. 16.

Uživatel se může v systému přihlásit na jednotlivé události, stahovat soubory u automobilů a vytvořit si rezervaci automobilu.

Vyšší pravomoc má v informačním systému *Technik*, který má navíc oproti *Uživateli* právo vytváření nových automobilů včetně QR kódu, spravovat tankovací karty, nahrávat soubory u automobilů. Má také pravomoc upravovat historii vytvořených tankovacích karet a historii vytvořených automobilů.

Nejvyšší pravomoc má v informačním systému *Administrátor*, který má veškerá práva jako *Uživatel* a *Technik* a má navíc pravomoc vytvořit a spravovat události a vytvořit nové tankovací karty.



Obrázek 16 - Use Case diagram

Pro některé z vybraných aktivit bude níže uveden scénář, který bude sloužit k lepšímu pochopení konkrétních skutečností. Tyto scénáře budou zpracovány podle následujícího vzoru:

Use Case	UCX – Název
Popis	Krátký popis konkrétního případu
Aktéři	Osoby spojené s daným případem
Podmínky pro spuštění	Podmínky, které je nutno splnit před zahájením daného případu
Základní scénář	Seznam kroků ke konkrétnímu případu
Alternativní scénář	Seznam alternativních kroků
Podmínky ukončení	Podmínky ukončení při splnění daného případu

Tabulka 1 - Vzorový scénář

Use Case	UC1 - Přihlášení na událost
Popis	Přihlášení uživatele na události
Aktéři	Administrátor, technik, uživatel, systém
Podmínky pro spuštění	Uživatel musí být do systému zaregistrován
Základní scénář	1) Uživatel spustí informační systém 2) Informační systém se zobrazí 3) Uživatel klikne na záložku *Události* 4) Systém zobrazí seznam událostí 5) Uživatel si vybere konkrétní událost, na kterou se chce přihlásit 6) Uživatel klikne na *Přihlásit se* 7) Systém uživatele přihlásí na událost
Alternativní scénář	6a) Nelze se přihlásit, hláška „Dosažený maximální počet osob“
Podmínky ukončení	Uživatel je úspěšně přihlášen

Tabulka 2 - Přihlášení na událost

Use Case	UC1 – Nahrávání souborů
Popis	Nahrávání souborů do informačního systému
Aktéři	Systém, administrátor, technik, uživatel
Podmínky pro spuštění	Uživatel musí být do systému zaregistrován
Základní scénář	1) Uživatel spustí informační systém 2) Informační systém se zobrazí 3) Uživatel klikne na záložku *Vyhledávání* a záložku *Vozy* 4) Systém zobrazí vozy 5) Uživatel klikne na určitý vůz 6) Uživatel klikne na *Nahrávání souboru* 7) Systém spustí nahrávání souboru 8) Uživatel vybere soubor k nahrávání 9) Uživatel nahraje soubor 10) Systém zobrazí nahraný soubor a přidá ho do seznamu nahraných souborů
Alternativní scénář	9a) Systém vyhodnotí chybu
Podmínky ukončení	Systém úspěšně nahraje soubor

Tabulka 3 - Nahrání souboru

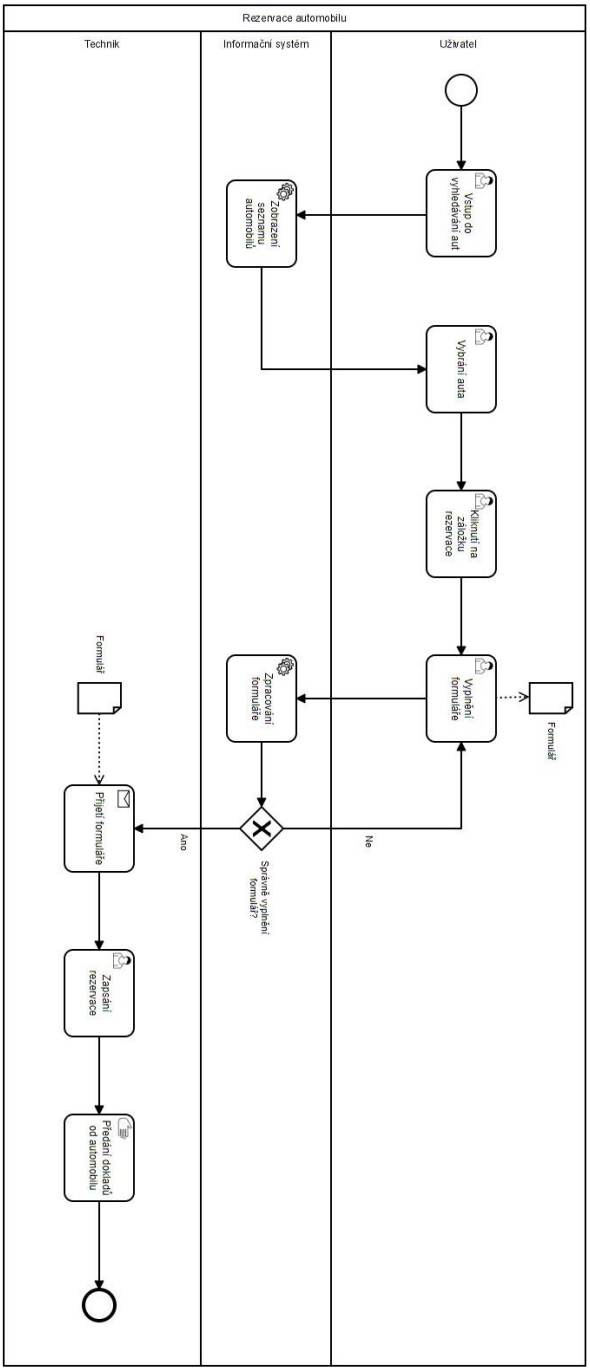
4.1.2.2 Business Process Model and Notation

Na níže uvedeném obrázku č. 17 můžeme vidět přesný postup rezervace automobilu. Krok po kroku vidíme různé alternativy, které mohou nastat při vyplňování rezervačního formuláře.

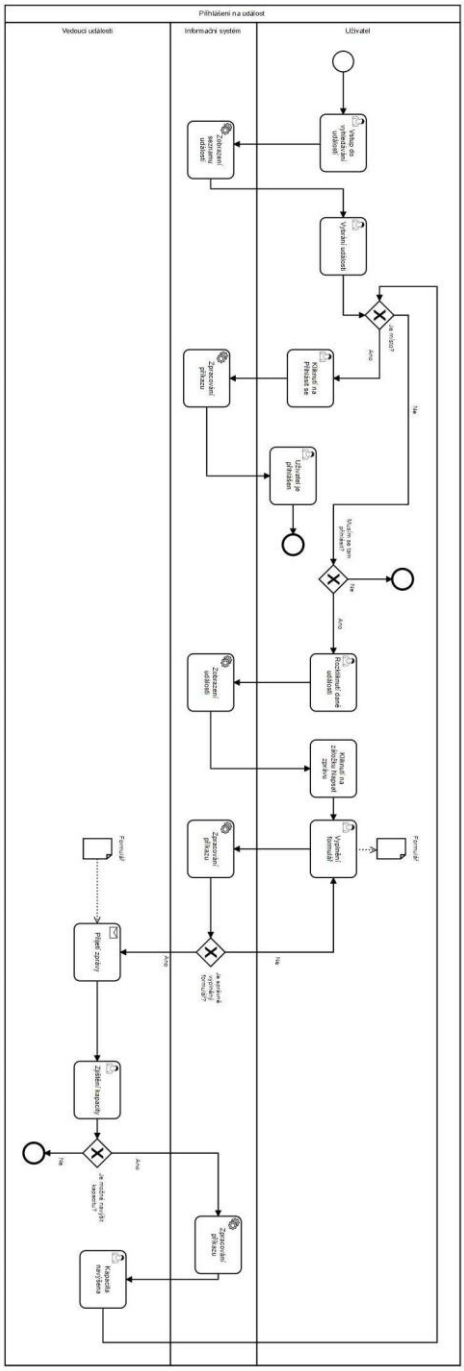
Nejdříve se uživatel přihlásí do informačního systému, kde si vyhledá potřebný automobil ze seznamu všech automobilů. U konkrétního automobilu klikne na záložku *Rezervace*, kde si vyplní formulář. Vyplněný formulář poté informační systém zpracuje. Pokud není formulář správně vyplněn, bude ho uživatel muset opravit. Pokud je formulář vyplněn správně, systém ho zašle technikovi automobilů. Ten zapíše rezervaci do kalendáře a předá uživateli doklady od daného automobilu.

Na dalším obrázku č. 18 můžeme vidět přesný postup přihlašování na událost. Také je zde znázorněn přesný postup a jeho různé alternativy, které mohou nastat při vyplňování přihlašovacího formuláře.

Uživatel se přihlásí do informačního systému, kde si vyhledá danou událost ze seznamu všech událostí, na kterou se chce přihlásit. Pokud ještě není zaplněná kapacita, klikne na *Přihlásit se*. Pokud je již kapacita zaplněna, může zaslat zprávu vedoucímu události. Uživatel vyplní formulář. Pokud není formulář správně vyplněn, bude ho uživatel muset opravit. Pokud je formulář vyplněn správně, systém ho zašle vedoucímu události. Ten po přijetí zprávy zjistí stav kapacity. Pokud kapacita půjde navýšit, navýší ji a uživatel se bude moci přihlásit. Proces však musí uživatel absolvovat od začátku.



Obrázek 17 - Rezervace automobilu



Obrázek 18 - Událost

4.1.2.3 Diagram tříd

Jako další oblastí diagramu uvádím diagram tříd, který slouží jako návod pro programátora při zpracování systému. Diagram tříd obsahuje jedenáct tříd, které jsou rozděleny na čtyř hlavních částí. První část se zabývá rozdělením pravomocí. Zde jsou uvedeny třídy *Uživatel*, *Admin* a *Technik*, kde se jednotlivě specifikují jejich pravomoci. *Admin* má pravomoc vytvořit, upravit a smazat událost, smazat osobu z události, vytvořit, upravit a smazat tankovací kartu, vytvořit QR kód k vozidlu, půjčit tankovací kartu a nahrávat soubory do systému. *Technik* má pravomoc nahrávat soubory do systému, upravovat a mazat tankovací karty, vytvořit QR kód k automobilu a půjčit tankovací kartu. *Admin, Technik a Uživatel* mají stejné pravomoc a to, přihlásit se a odhlásit se k událostem, rezervovat si vozidlo, vypůjčit si tankovací kartu a stahovat soubory v systému.

Druhá část je rozdělena na *Seznam uživatelů v události*, *Události* a *Seznam Událostí*. V *Seznamu uživatelů v události* najdeme seznam přihlášených osob. *Událost* je specifikována pomocí názvu události, místem konání, maximálním počtem osob, obsazeností, popisem události, vedoucím události a maximálním přihlašovacím a odhlašovacím datem. V *Seznamu událostí* najdeme seznam všech událostí.

Třetí část je věnována automobilům. Nalezneme zde třídu *Automobil* a *Evidence výpůjček automobilů. Automobil* je specifikován pomocí modelu, barvy, infotainmentem, dodatečnými informacemi, kalendářem, oddělením, které auto vlastní, druhem paliva, parkovací pozicí, počtem najetých kilometrů, druhem převodovky, propojením (WiFi připojení, USB připojení, Bluetooth aj.), QR kódem, rokem výroby, správcem, SPZ vozu, VINem, výbavou a značkou vozu. Třída *Evidence výpůjček automobilů* je specifikována pomocí data vypůjčení a vrácení a stavem.

Čtvrtá část je věnována tankovacím kartám a je rozdělena na třídu *Tankovací karta* a *Evidence tankovacích karet. Tankovací karty* jsou specifikovány pomocí čísla karty, správcem karty, PINem a stavem karty. Třída *Evidence tankovacích karet* je specifikována pomocí data vypůjčení a vrácení a stavem (volný/zapůjčeno).

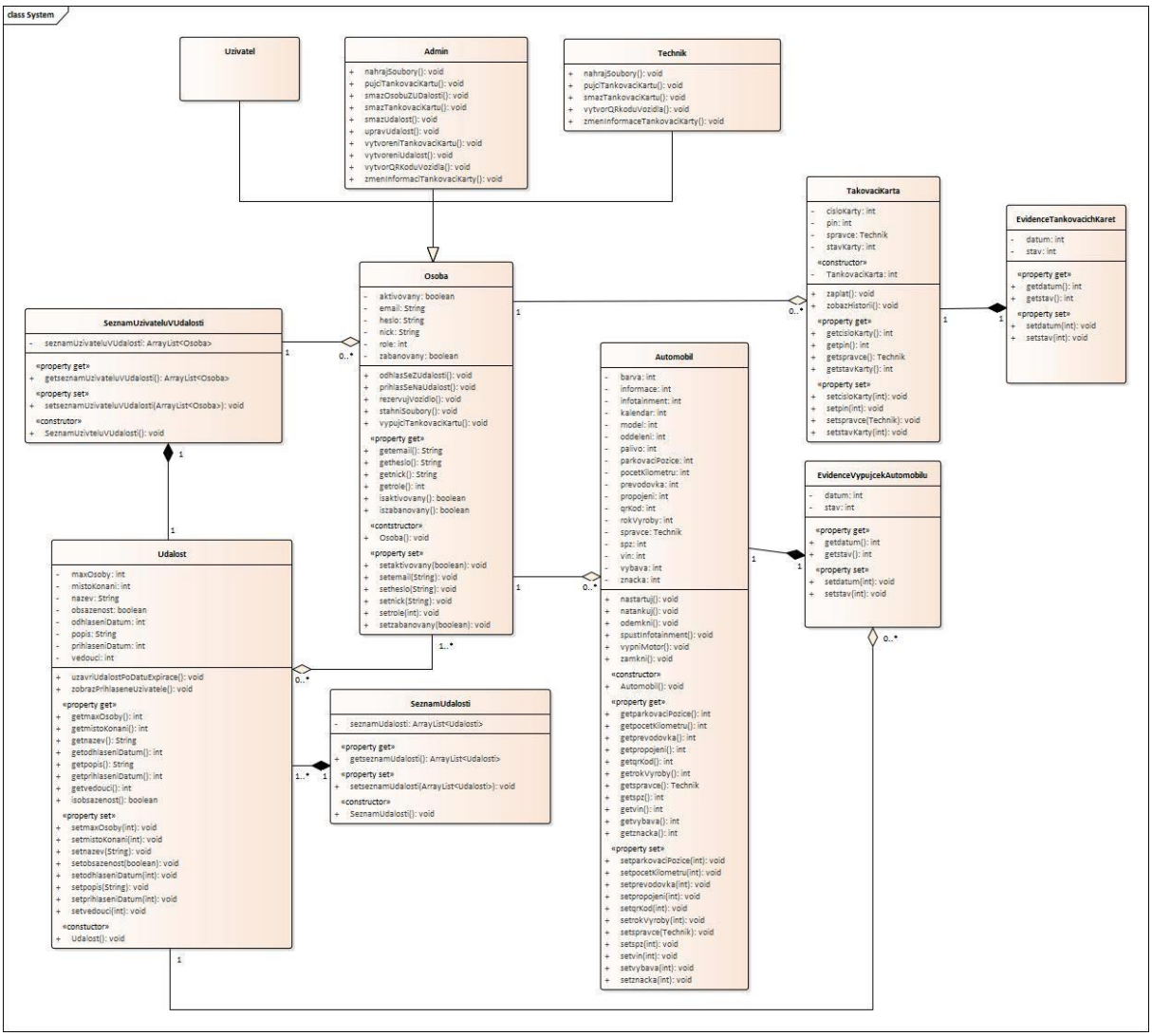
Pro třídy *Uživatel, Admin* a *Technik* je nadtřídou třída *Osoba*, která zajišťuje, že společné údaje nebudou zapsány na více místech. Tento vztah se nazývá generalizace. Třída *Osoba* je specifikována nickem osoby, heslem, rolí v systému, e-mailem, aktivovaným či zabanovaným účtem v systému.

Třídy *Tankovací karta*, *Automobil*, *Událost* a *Seznam uživatelů v události* uvádí vzájemný vztah k třídě *Osoba*. Tento vztah nazýváme agregace. Třída *Událost* má vztah k třídám *Seznam uživatelů v události* a *Seznam událostí*. Tento vztah nazýváme kompozice.

Třída *Evidence výpůjček automobilů* je svázána pomocí vztahu kompozice s třídou *Automobil.*

Třída *Evidence tankovacích karet* je svázána pomocí vztahu kompozice s třídou *Tankovací karta.*

Abychom podrobněji popsali vztahy na diagramu tříd, uvádím příklad multiplicity na třídách *Osoba* a *Tankovací karta*. Jedna *Osoba* může mít 0-n tankovacích karet, ale *Tankovací karta* může být zapůjčena pouze jedné *Osobě.* Všechny ostatní multiplicity jsou vyobrazeny na níže uvedeném diagramu.



Obrázek 19 - Diagram tříd

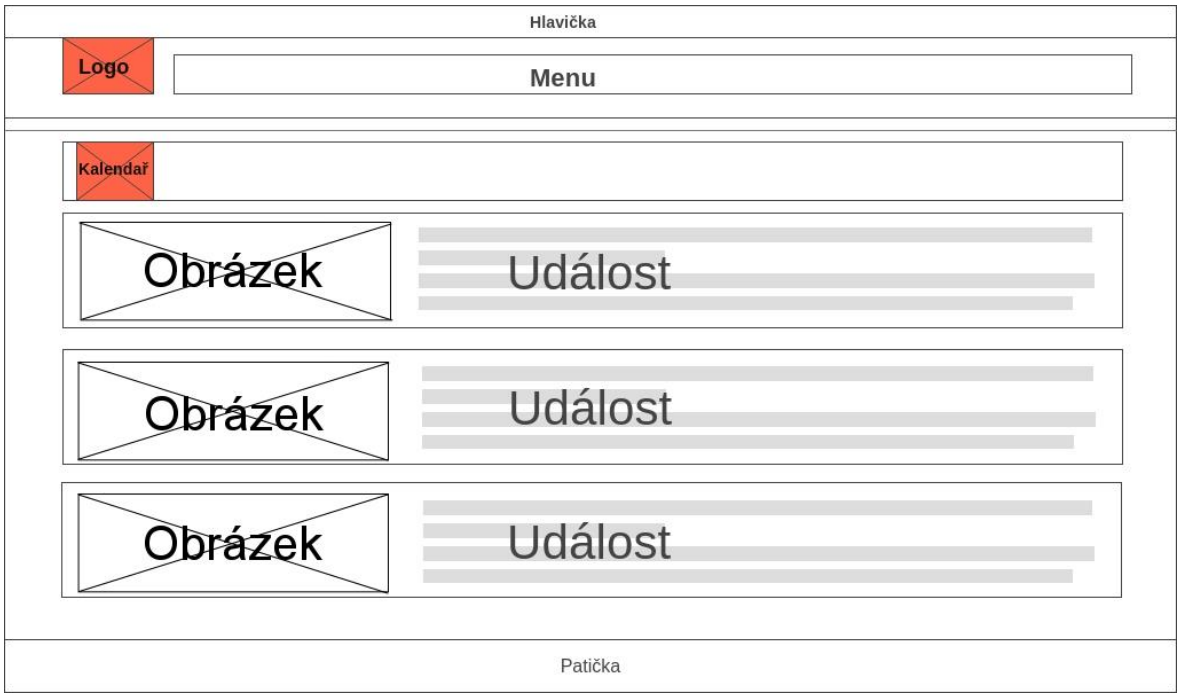
4.2 Návrh

Pro návrh informačního systému byl využit Wireframe. Wireframe je skica stránky, jejímž cílem je ukázat, jak by cílová stránka měla vypadat.

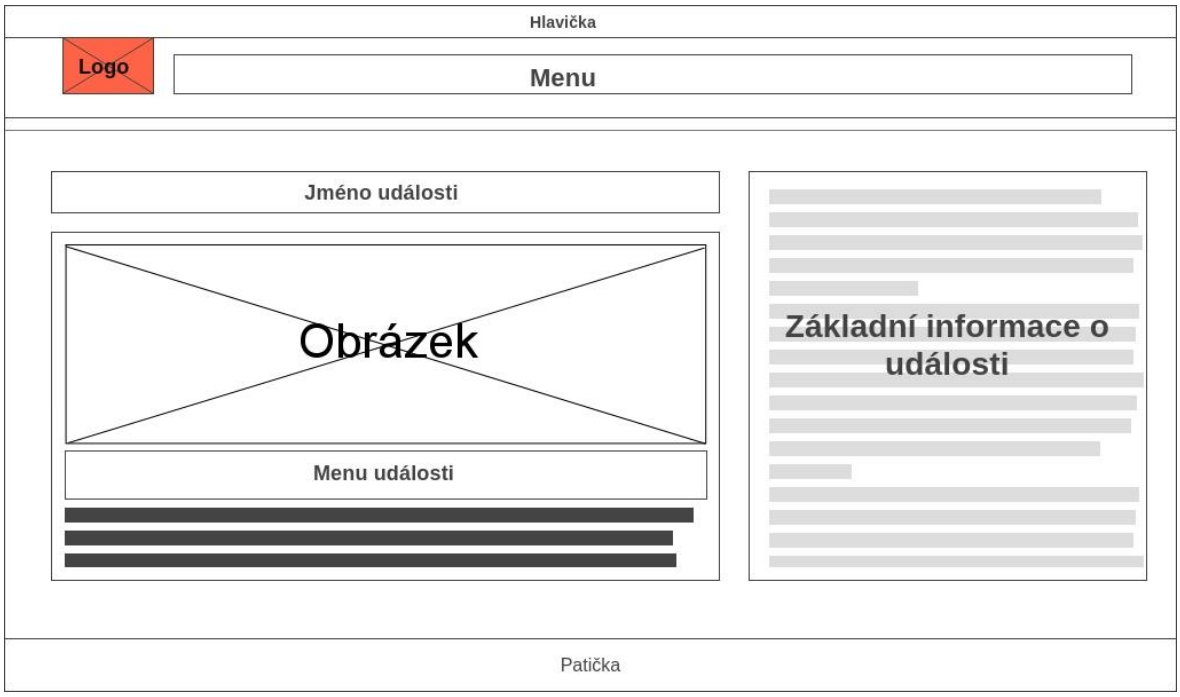
4.2.1 Wireframe

Wireframe webu nám slouží, jak již bylo výše zmíněno, jako takzvaná skica webové stránky, která zobrazuje rozmístění funkcí a obsah webové stránky. Wireframe nezahrnuje žádné grafické prvky a postrádá vizuální estetiku. Je tvořen pouze pomocí čar a textu.

Na níže uvedeném obrázku demonstruji Wireframe na konkrétním příkladu, a to na seznamu událostí a zobrazení konkrétní události. Je zde také uvedeno již původní menu s rozšířením.



Obrázek 20 - Wireframe: seznam událostí



Obrázek 21 - Wireframe: událost

Na obrázku č. 21 je vyobrazena struktura příkladové události, která je tvořena jménem události, základními informacemi o události a menu události.

4.3 Implementace

Následující kapitoly budou zaměřené, jak již název napovídá, na implementaci rozšíření manažerského informačního systému. V jednotlivých kapitolách bude ukázaný jak vzhled implementované funkce, tak i části kódu k příslušné funkci.

Stránka se nachází na doméně www.wasah.cz. Poskytovatel webhostingu je firma Endora.cz, kde najdeme solidní cenu hostingu a pro chod webových stránek dostatečné možnosti.

4.3.1 Databáze

Při programování informačního systému je zapotřebí ukládat informace do databáze. Jako databázi jsem využil databázi MySQL, která je spravována programovým systémem phpMyAdmin. Databázi MySQL jsem vybral z toho důvodu, jelikož jsem informační systém programoval v programovacím jazyce PHP. Ten s touto databází velmi dobře spolupracuje a umožňuje rychlý přístup k uloženým datům. Aby se na každé stránce,

která spolupracuje s databází, nemusela zadávat vždy informace k přístupu do databáze, je zde udělen příkaz *include_once ("databaze/pripojit.php").* Ten si načte ze souboru *pripojit.php* veškeré potřebné údaje pro připojení se k databázi.

Níže uvádím obrázek č. 22, kde je vyobrazen seznam použitých tabulek.

Tabulka	Operace	Řádků	Typ	Porovnávání	Velikost	Navíc
AA_aplikace	Projít Struktura Vyhledávání Vložit Vyprázdnit Odstranit	24	MyISAM	utf8_general_ci	4,6 KiB	-
auta	Projít Struktura Vyhledávání Vložit Vyprázdnit Odstranit	11	MyISAM	utf8_general_ci	10,6 KiB	-
auta_historie	Projít Struktura Vyhledávání Vložit Vyprázdnit Odstranit	102	MyISAM	utf8_general_ci	8,6 KiB	-
CarPlay_aplikace	Projít Struktura Vyhledávání Vložit Vyprázdnit Odstranit	19	MyISAM	utf8_general_ci	6,1 KiB	-
CarPlay_telefony	Projít Struktura Vyhledávání Vložit Vyprázdnit Odstranit	7	MyISAM	utf8_general_ci	2,5 KiB	-
event	Projít Struktura Vyhledávání Vložit Vyprázdnit Odstranit	3	MyISAM	utf8_general_ci	2,5 KiB	144 B
event_registrace	Projít Struktura Vyhledávání Vložit Vyprázdnit Odstranit	21	MyISAM	utf8_general_ci	2,9 KiB	68 B
lidi	Projít Struktura Vyhledávání Vložit Vyprázdnit Odstranit	12	MyISAM	utf8_general_ci	3,3 KiB	-
mirolink_aplikace	Projít Struktura Vyhledávání Vložit Vyprázdnit Odstranit	59	MyISAM	utf8_general_ci	11,8 KiB	-
mirolink_telefony	Projít Struktura Vyhledávání Vložit Vyprázdnit Odstranit	54	MyISAM	utf8_general_ci	6,9 KiB	-
online_prehled	Projít Struktura Vyhledávání Vložit Vyprázdnit Odstranit	13	MyISAM	utf8_general_ci	1,9 KiB	40 B
tank_historie	Projít Struktura Vyhledávání Vložit Vyprázdnit Odstranit	125	MyISAM	utf8_general_ci	10,6 KiB	-
tank_k	Projít Struktura Vyhledávání Vložit Vyprázdnit Odstranit	4	MyISAM	utf8_general_ci	2,9 KiB	480 B
telefony	Projít Struktura Vyhledávání Vložit Vyprázdnit Odstranit	51	MyISAM	utf8_general_ci	16,3 KiB	-
uzivatele_prehled	Projít Struktura Vyhledávání Vložit Vyprázdnit Odstranit	21	MyISAM	utf8_general_ci	3,5 KiB	40 B
15 tabulek	Celkem	526	MyISAM	utf8_general_ci	94,9 KiB	772 B

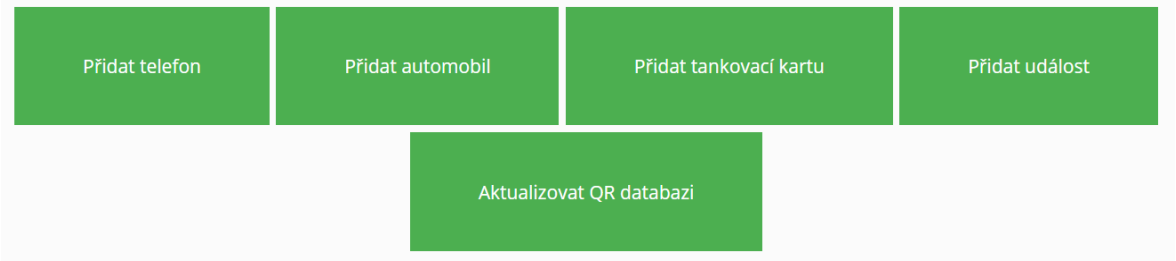
Obrázek 22 - Seznam použitých tabulek v databázi MySQL

Pro představu uvádím kód, co obsahuje soubor *pripojit.php*.

```
<?php
 $link = mysqli_connect("localhost", "nick", "heslo", "jméno databaze");
 if (!$link) {
    echo "Error: Unable to connect to MySQL." . PHP_EOL;
    echo "Debugging errno: " . mysqli_connect_errno() . PHP_EOL;
    echo "Debugging error: " . mysqli_connect_error() . PHP_EOL;
    exit();
 } ?>
```

4.3.2 Přidávání nových položek

Do stávajícího systému jsem přidal několik nových položek, jako je *Přidat tankovací kartu, Přidat událost* a *Aktualizovat QR databázi.* V položkách *Přidat tankovací kartu* a *Přidat událost* má pravomoc zadávat nové položky pouze administrátor. V položce *Aktualizovat QR databázi* může pracovat administrátor i technik. V informačním systému každá osoba (administrátor, technik a uživatel) vidí pouze ty položky, ke kterým má oprávnění (tj. uživatel nevidí žádnou z těchto položek).



Obrázek 23 – Tlačítka

Programovací kód pro tlačítko *Přidat událost*:

```
<button  name="pridej" type="submit" value="udalost" style="background-color: #4CAF50; border: none;  color: white; padding: 2% 3%; text-decoration: none;margin: 4px 2px;cursor: pointer;font-size : 150%;"> Přidat událost  </button>
```

Skript funkce přidání události:

```
}
if  ($_GET["pridej"]=="udalost")     {echo "<h2 style='color: green'><p>Událost byla vytvořena.</p></h2>";
  include_once ("databaze/pripojit.php");
  $dotaz = "INSERT INTO event (nazev, popis, obsazeno, max_osoby, vedouci, prihlaseni_datum, odhlaseni_datum, misto_konani) VALUES ('Nová událost', '', '', '', '', '', '', '')";
  $vysledek = mysqli_query($link, $dotaz);
  if(!$vysledek) {
    echo "error pridej udalost.php";    exit();  }   include_once ("databaze/odpojit.php");
```

4.3.2.1 Tankovací karty

Vzhledem k tomu, že na oddělení je méně tankovacích karet, než je testovacích automobilů, bylo potřeba zavést nějakou inovaci ve formě správy karet. Tankovací karty slouží k tankování automobilů při služebních či testovacích cestách. Zaimplementoval jsem je do systému proto, aby bylo možné rychle dohledat, kdo jí má zrovna zapůjčenou a aby byl přehlednější finanční tok.

Díky této funkci bude viditelná denní útrata. Maximální denní částka je stanovena na 4.000 Kč.

Pro lepší představu uvádím část PHP kódu, který nás u každé tankovací karty informuje, zda je karta zapůjčená či nikoliv:

```
<?php
```

if ($radek->zapujcil_jmeno == "vráceno") { echo "Karta k zapůjčení"; }

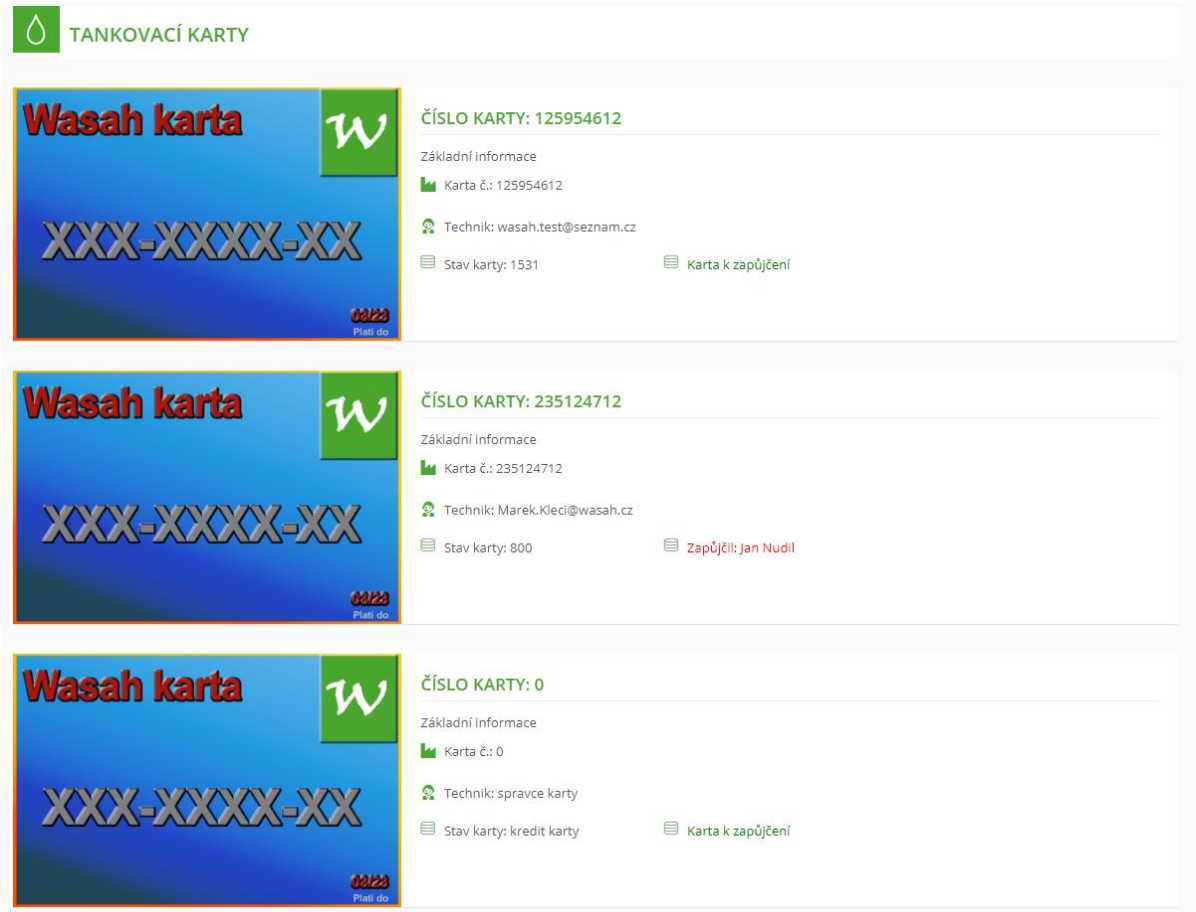
else { ?>

<?php echo "Zapůjčil: ".$radek->zapujcil_jmeno; ?>

<?php } ?>

Na níže uvedeném obrázku (č. 24) jsou znázorněné tři tankovací karty. První dvě jsou dle vyžadovaných kritérií správně vyplněné. Jsou u nich uvedená jejich čísla, kdo je technikem dané karty a jaký je jejich stav, jelikož, jak již bylo zmíněno, každá tankovací karta má stanovený vlastní denní limit.

Třetí z uvedených karet nemá vyplněné hodnoty, jelikož se jedná o nově vytvořenou kartu v systému a je třeba ještě její hodnoty vyplnit. Tankovací karty může do systému přidávat pouze administrátor nebo technik systému.



Obrázek 24 - Seznam tankovacích karet

Po rozkliknutí vybrané tankovací karty můžeme vidět záložku *Číslo karty*. Dále zde máme záložku *Technik*, která nám říká, kdo je za danou kartu a její správu zodpovědný. Dalšími jsou záložky *Zapůjčil, Zapůjčil e-mail, Zapůjčil telefon.* Záložka *Zapůjčil* nám říká, kdo danou tankovací kartu vypůjčil. Pokud je tankovací karta volná, je zde vyplněno heslo „vraceno“. Záložka *Zapůjčil e-mail* nám udává e-mail na osobu, která si danou kartu zapůjčila. Záložka *Zapůjčil telefon* nám udává telefonní číslo na osobu, která si danou tankovací kartu zapůjčila. Poslední záložkou je *Stav karty*, pod kterou se skrývá informace, kolik peněz je na tankovací kartě ještě k dispozici.

V menu tankovací karty je záložka *Historie*. Tato záložka pod sebou skrývá odkaz na PDF soubor, ve kterém je evidována historie dané karty (tj. historie zapůjčení i historie plateb). O této části se více dozvíme v další kapitole.

Obrázek 25 - Vzhled karty

4.3.2.1.1 Převod do PDF

Další novou funkcí, kterou jsem implementoval v novém rozšíření, je generování historie zapůjčení tankovacích karet a historie zapůjčení automobilů. Historii v PDF

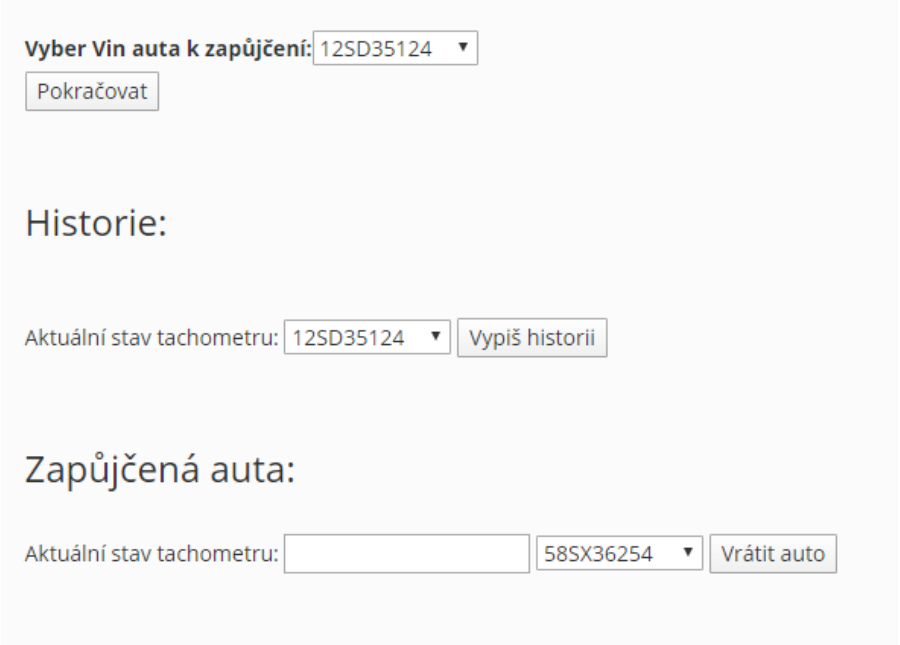
souboru si mohou zobrazit libovolní uživatelé informačního systému. Ze souboru PDF k historii tankovacích karet můžeme vyčíst informace jako číslo karty, disponibilní zůstatek na kartě, denní útrata, kdo má kartu zapůjčenou, datum jejího zapůjčení a kontakt na osobu, která tankovací kartu zapůjčila. Soubor také slouží k uchování historie karet.

Ze souboru PDF k historii zapůjčení automobilů můžeme vyčíst informace jako VIN automobilu, datum vypůjčení, kdo má automobil zapůjčený, kdo automobil zapůjčil, stav najetých kilometrů a informaci, zda byl automobil zapůjčen k nějaké události.

Historie karty

ID	C_karty	Stav	Utrata	Zapujcil	Datum	Zapsal
129	789456314	4000	0	vraceno	06. 04. 2020	Tomas.Marny@wasah.cz
130	789456314	4000	0	PetrKlic@wasah.cz	06. 04. 2020	Tomas.Marny@wasah.cz
131	789456314	1253	-2747	vraceno	06. 04. 2020	Tomas.Marny@wasah.cz

Obrázek 26 - PDF u tankovací karty

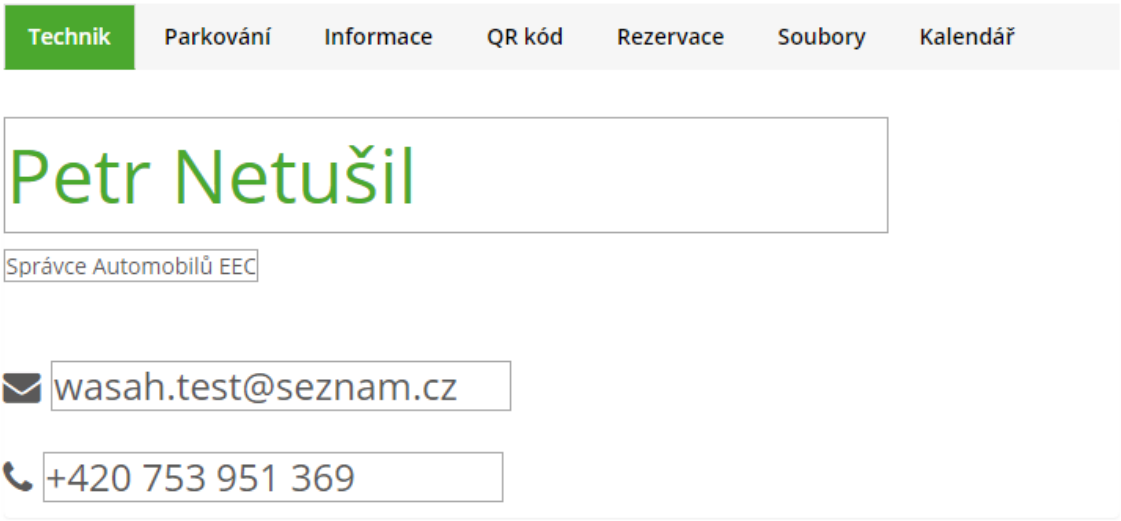


Obrázek 27 - Správa historie automobilu

4.3.2.2 Správa automobilů

Správa automobilů dostala nové záložky jako je *QR kód, Rezervace, Soubory a Kalendář*. Každou nově přidanou záložku budu popisovat v dalších kapitolách.

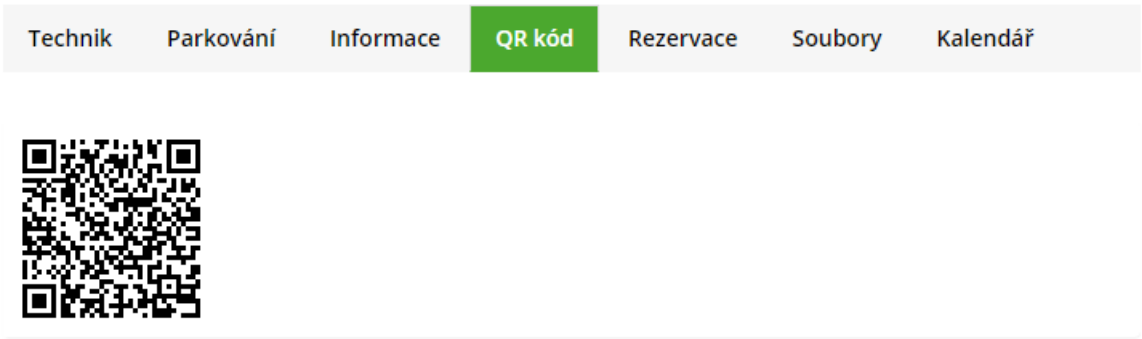
Administrátor může v informačním systému veškeré změny editovat. Kdežto technik může změny provádět pouze u automobilů, ke kterým je přiřazen.



Obrázek 28 - Záložky u automobilu

4.3.2.2.1 QR kód

V dnešní době, kdy jsou uživatelé velmi nároční a chtějí mít vše rychle dohledatelné a nejlépe ihned při ruce, se hodí mít pro tyto účely možnost vytvoření QR kódu, který uživatel naskenuje a bude odkázaný na e-kartu vozu bez zbytečného a zdlouhavého hledání daného vozidla. QR kód můžeme přilepit na desky automobilu nebo přímo na libovolnou část automobilu.



Obrázek 29 - QR kód

4.3.2.2.2 Kalendář

Kalendář obsahuje jednu z důležitých informací a to, zda je automobil volný nebo vypůjčený. Dříve byly všechny kalendáře na jedné samostatné stránce, ale nově byly přesunuty vždy ke konkrétnímu automobilu, což ulehčí zjišťování dostupnosti automobilu.

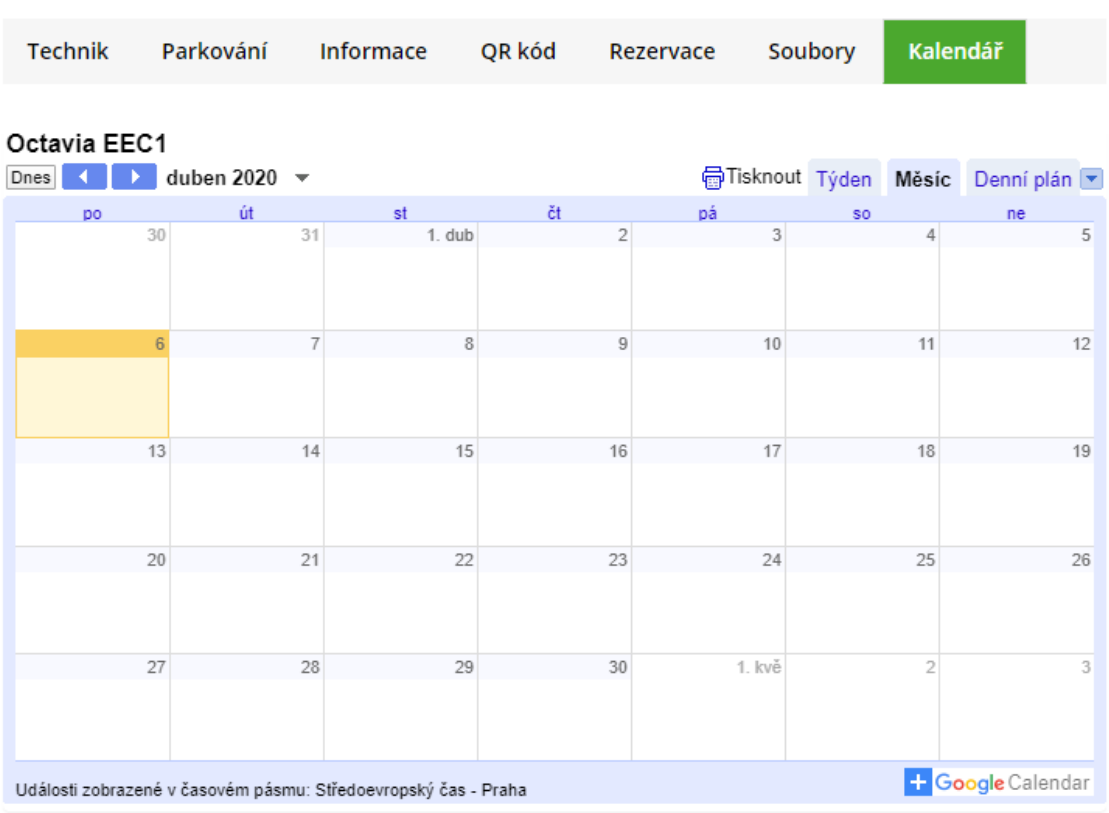
Pravomoc na spravování odkazu na kalendáře má technik i administrátor. Pro uživatele je možnost upravovat/vidět odkaz skryta.

Obrázek 30 - Link na kalendář

PHP kód nám umožnuje z databáze načítat odkaz kalendáře, což nám umožňuje mít u každého automobilu vlastní kalendář.

Kód na stránce je následující:

```
<div class="tab-pane fade features" id="tab7default">
<iframe src="<?php echo $radek->kalendar; ?>"
style="border: 0" width="800" height="600" frameborder="0"
scrolling="no"></iframe></div>
```

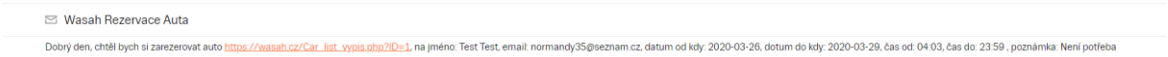


Obrázek 31 – Kalendář

4.3.2.2.3 Rezervace

Aby nemusel každý zájemce o automobil psát speciální e-mail s žádostí, nebo aby nemusel telefonovat technikovi automobilů, byla pro tyto účely vytvořena kategorie *Rezervace*, kterou zájemce o automobil vyplní a následně odešle technikovi. Samozřejmě se zde předpokládá, že osoba, která vypisuje rezervační formulář, se předem podívala na záložku kalendář, aby nedocházelo ke konfliktu termínů. Po odeslání správně vyplněného formuláře přijde technikovi e-mail, kde budou zobrazené informace, se kterýma bude dále pracovat.

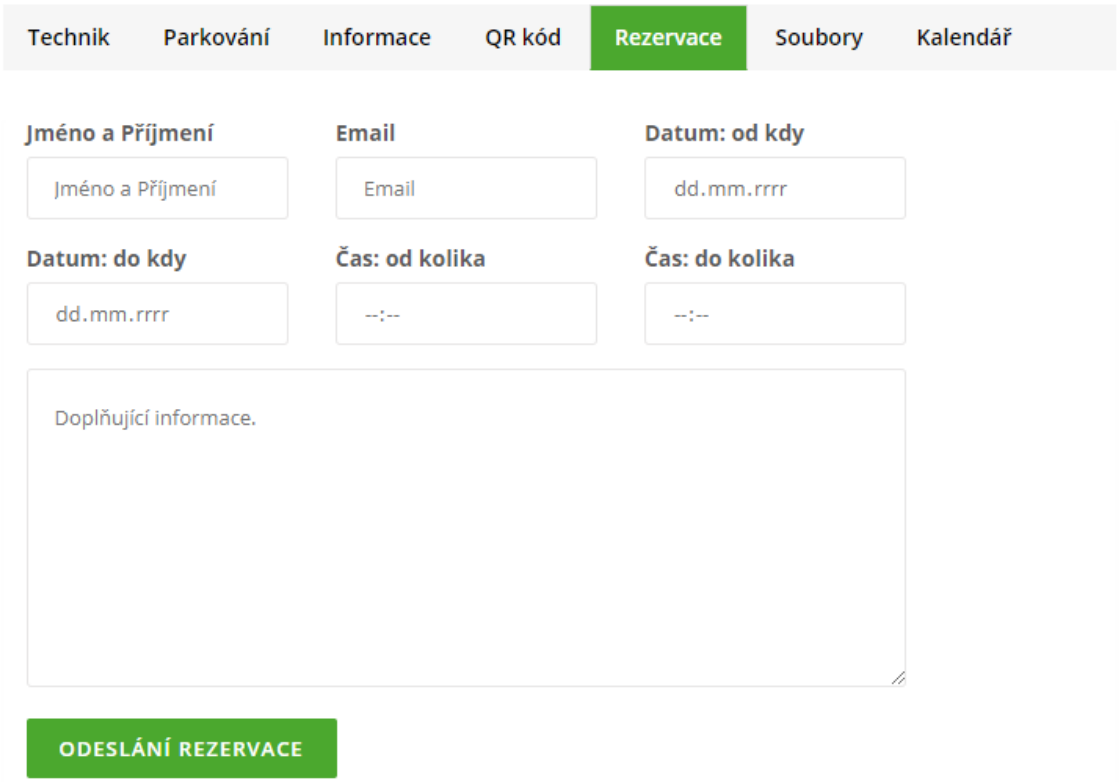
Vzhled e-mailu, který přijde technikovi:



Obrázek 32 – E-mail technikovi

Kód formuláře rezervace:

```
<form action="rezervace_auta.php" method="GET">
 <div class="col-lg-4 col-md-4 col-xs-12">
        <label for="name">Jméno a Příjmení</label>
                <div class="form-group fullname">
<input type="text" name="jmeno" class="input-text" id="fullname" placeholder="Jméno a Příjmení"></div></div>
                <div class="col-lg-4 col-md-4 col-xs-12">
                        <label for="name">Email</label>
                        <div class="form-group fullname">
<input type="email" name="email" class="input-text" id="fullname" placeholder="Email">
<input type="hidden" name="email_spravce" value="<?php echo $radek->odd_email;?>">
<input type="hidden" name="odkaz" value="<?php echo $radek->ID;?>">
                        </div>
                                ...
                        </div> </form>
```



Obrázek 33 - Rezervace automobilu

4.3.2.2.4 Nahrání a stažení dokumentů

Nahrávání a stahování souborů jsem do informačního systému zaimplementovat proto, jelikož se během práce s automobily dané vozy upravují dle potřeby. Proto je třeba, aby všechny zainteresované osoby měly k těmto dokumentům přístup. Tyto dokumenty nám ukazují jak původní konfiguraci a parametry vozidla, tak také slouží pro různé aktualizace infotainmentu. Slouží také i k evidenci poškozených částí vozidla, popřípadě nefunkčnosti jakékoliv části vozidla.

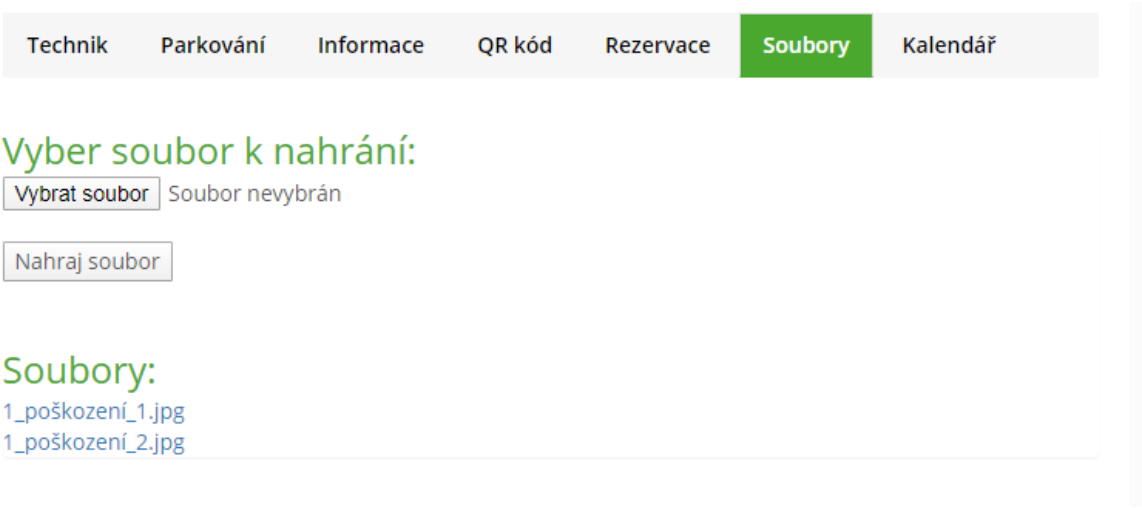
Do informačního systému lze nahrávat soubory jak k daným automobilům, tak i k událostem, které se již konaly či se teprve budou konat a je zapotřebí se na ně dostatečně připravit.

Do informačního systému je možné nahrávat soubory či obrázky. Obrázky musí být ve formátu grafického souboru (.jpg, .jpeg apod.). Soubory též musí být ve formátu běžných souborů. Lze nahrát soubor .pdf (univerzální formát), a dále od firmy Microsoft .doc/x (textový soubor), .ppt/x (prezentace), .xls/x (soubor Excel) aj.. Dá se říci, že se dá

nahrát jakýkoliv typ souboru, který se předem v informačním systému nadefinuje. Veškeré soubory se dají stáhnout.

Část kódu, který pomáhá ke správě souborů:

```
<form action="files.php" method="post" enctype="multipart/form-data">
   <font size="5" color="#4BA82E">Vyber soubor k nahrání: </font>
   <input type="file" name="fileToUpload" id="fileToUpload"><br>
   <input type="submit" value="Nahraj soubor" name="submit">
<input id="prodId" name="prodId" type="hidden" value="<?php echo $radek->ID?>">
</form>
<br><br><font size="5" color="#4BA82E">Soubory:</font> <br>
                    <?php
                        $files = scandir( "files" );
                        foreach( $files as $file ){
                          if ($file === '.' or $file === '..') continue;
                          if  (preg_match('#'.$radek->ID.'_#',$file)){echo  "<a  href='files/$file'
download>$file</a><br>";}  }?> </div>
```

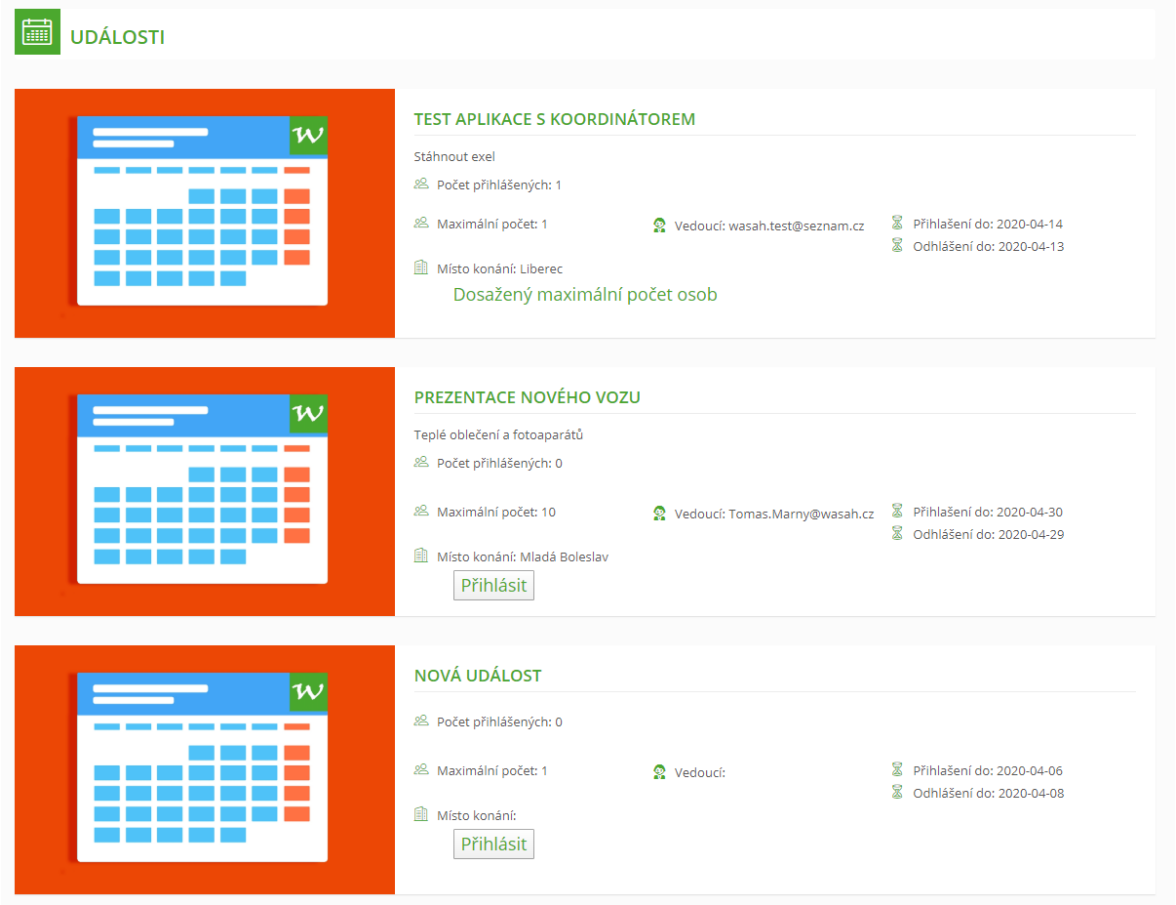


Obrázek 34 – Soubory

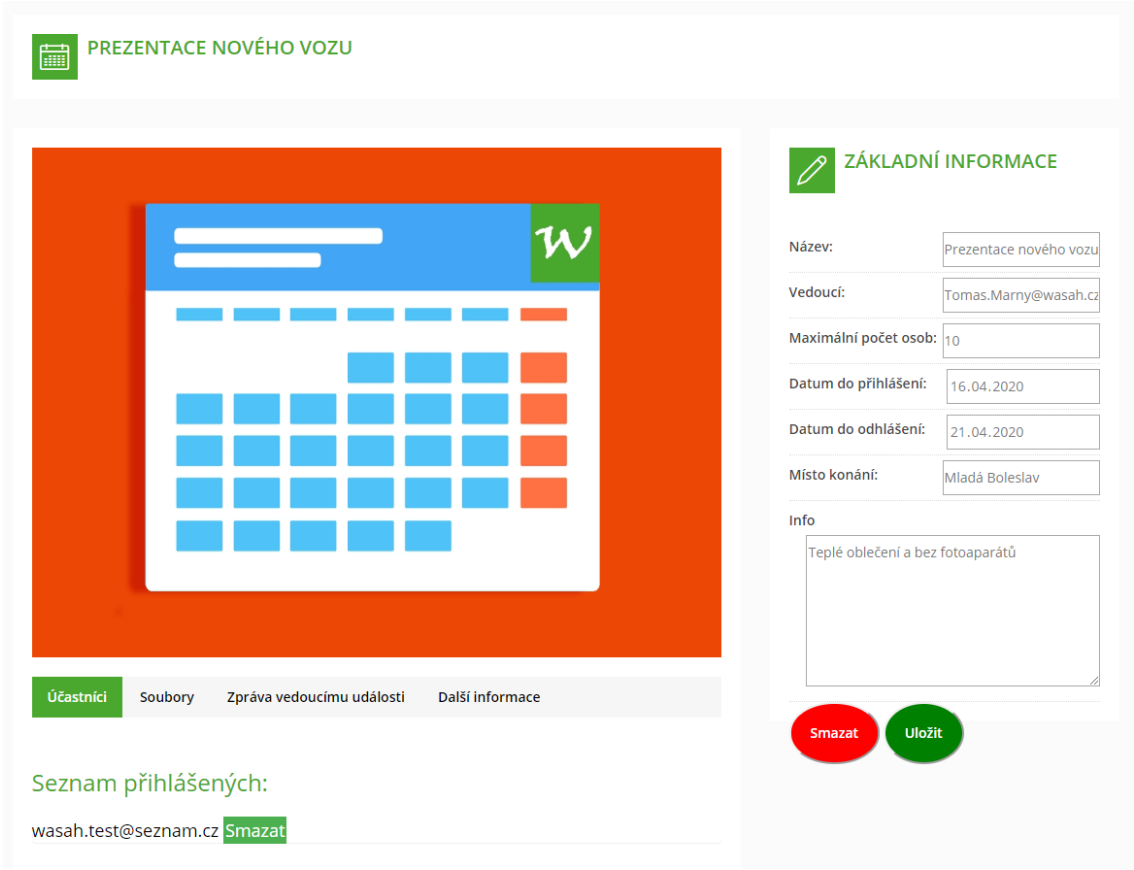
4.3.2.3 Události

Událost je nová záložka v informačním systému, která byla vytvořena pro rychlejší komunikaci mezi jednotlivými účastníky. Tato záložka byla vyžadována hlavně ze strany vedení, jelikož bylo velmi složité informovat všechny zainteresované osoby v rámci vnitropodnikové pošty. Nyní má každý uživatel systému možnost se na danou událost ihned přihlásit a ví, zda se může události zúčastnit. Dříve musel žadatel počkat na odezvu v rámci vnitrofiremní pošty.

U události máme vyobrazeny všechny konkrétní specifikace - název události, vedoucí události, maximální počet přihlášených osob, místo konání a další, konkrétnější informace o události. Je zde také informace o maximálních datech možného přihlášení a odhlášení.



Obrázek 35 – Událost



Obrázek 36 - Událost

PHP kód pro vypsání přihlášených uživatelů:

```
 <?php
$dotaz_vypis="SELECT * FROM event_registrace WHERE id_udalosti = '$radek->ID'";
                $vysledek_vypis = mysqli_query($link, $dotaz_vypis);
                if(!$vysledek_vypis) {
                  echo "error vypis prihlasenych";
                 exit();                        }
               $pocet_vypis = mysqli_num_rows($vysledek_vypis);
                for ($i_vypis = 0; $pocet_vypis > $i_vypis; $pocet_vypis--) {
                      $radek_vypis=mysqli_fetch_object($vysledek_vypis); ?>
                <font  size="4" color="black"> <?php echo $radek_vypis->email; ?>
                </font>
                <br>
               <?php } ?>
```

PHP kód pro možnost mazání uživatelů:

```
 <?php
if ($_SESSION["uzivatel"]->prava=='admin' || $_SESSION["uzivatel"]-
>prava=='superadmin') {    ?>
```

<a href="event_odhlaseni.php?email=<?php echo "$radek_vypis->email&event=$radek->ID"?>" OnClick="return confirm('Opravdu tohoto uživatele chceš smazat?');" class="button_smazat"> Smazat <?php } ?>

4.3.2.3.1 Zpráva vedoucímu události

Ačkoliv je u každé události, která je v systému zadána, uvedeno mnoho informací (místnost konání, maximální počet osob, koncová data přihlášení a odhlášení, vedoucí události či počet již přihlášených osob), může se stát, že osoba, která se chce na událost přihlásit, bude mít další doplňující otázky, které bude chtít zodpovědět. K tomu slouží záložka *Zpráva vedoucímu události*. Každá událost obsahuje formulář, pomocí kterého může kdokoliv kontaktovat zodpovědnou osobu s jakýmkoliv dotazem.

Obrázek 37 - Zpráva vedoucímu události

Kód zprávy vedoucímu události:

* *
* *

```
<h2 class="title">Zpráva vedoucímu události</h2>
<div class=" contact-form">
<form action="index.html" method="GET">
<div class="row">
<div class="form-group  col-xs-12 col-sm-12 col-md-12">
<label for="name">Jméno a Příjmení</label>
<input type="text" name="fullname" class="input-text" id="name" placeholder="">
</div>
 <div class="form-group  col-xs-12 col-sm-12 col-md-12">
 <label for="email">E-mail</label>
 <input type="email" name="email" class="input-text" id="email" placeholder="">
 </div>
 <div class="form-group  col-xs-12 col-sm-12 col-md-12">
 <label for="message">Zpráva</label>
 <textarea class="input-text" name="message" id="message" placeholder=""></textarea>
 </div>
 <div class="form-group  col-xs-12 col-sm-12 col-md-12">
 <button type="submit" class="btn btn-submit btn-block">Odeslat zprávu</button>
</div> </div> </form> </div>
</div> </div> </div>
```

4.3.3 Testování

V této části diplomové práce jsem se zabýval testováním, použitelností, přístupností a rychlostí webové stránky. Na testování webové stránky jsem využil nejpoužívanější internetové prohlížeče (Internet Explorer, Google Chrome a Mozilla Firefox). Po celou dobu vývoje jsem testoval jednotlivé části aplikace, tj. pokud jsem právě dokončil programování záložky U*dálosti* ve webové aplikaci, poté jsem testoval pouze tuto část. V této části jsem odstranil všechny chyby, na které jsem během testování narazil.

Další částí testování bylo simulování a chování jednotlivých uživatelů systému, kde jsem se též snažil eliminovat veškeré chyby. V této části jsem jich našel poměrně velké množství (například se mi částečně prolínaly pravomoci administrátora a technika).

Poslední částí testování bylo tzv. testování v praxi. Aplikace byla spuštěna na doméně www.wasah.cz. Do testování bylo zapojeno několik testerů ze strany kolegů. V této části testování byly doladěny poslední drobné chyby a dříve nezaregistrované a nevnímané detaily například ze strany grafické.

Chtěl bych zde zdůraznit, že jednotlivé testování je občas velmi opomíjená část při realizaci různých webů, portálů nebo informačních systémů. V nejideálnějším případě by se každý vytvořený systém měl otestovat v co největším počtu uživatelů. Čím více uživatelů daný systém testuje, tím je větší pravděpodobnost, že se odhalí i ty nejmenší chyby a nedostatky, které by se jinak přehlédli. Více testujících uživatelů poskytne také větší zpětnou vazbu a více námětů na úpravy či další rozšíření. Právě zpětná vazba je tím, co dá konečnou formu celému informačnímu systému.

4.3.3.1 Přístupnost

Pro otestování přístupnosti byla použita aplikace Wave accessibility tool na adrese: https://wave.webaim.org. Přístupnost webu musí být bezproblémová pro použití webové aplikace uživatelům se specifickým postižením. Proto byla testována dle metodiky WCAG 2.1 (Web Content Accessibility Guideline), vytvořené mezinárodní konsorciem W3C.

4.3.3.2 Rychlost

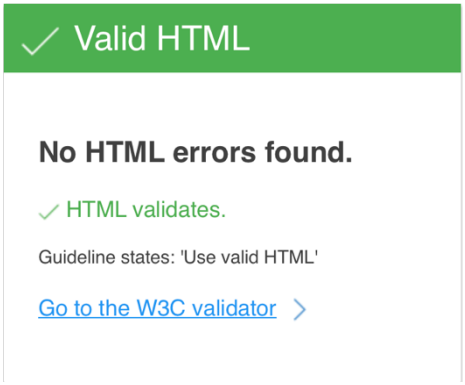
Rychlost webové stránky jsem otestoval pomocí online nástroje Google PageSpeed (https://developers.google.com/speed/):

- Tento nástroj testuje rychlost načítání webu jak pro desktopovou verzi, tak i pro mobilní zařízení
- V kategorii desktopu web získal skóre 88 a v kategorii mobilní zařízení dosáhl skóre 81

Dle výsledků obou testovacích nástrojů je web dostatečně rychlý. Vylepšení rychlosti by se dalo dosáhnout pomocí prodloužení času expirace cachovaných prvků. Problém s Render blocking CSS / JS by se dal vyřešit odstraněním pomocí funkce PageSpeed Insights od společnosti Google.

4.3.3.3 Validita kódu

Pro kontrolu zdrojových kódů byly použity služby validator.w3.org. Bylo použito k otestování validace HTML a CSS kódu. Výsledný validní HTML kód je mnohem přehlednější a čistší. Výsledek validátoru nám říká, že náš HTML kód je zcela validní.



Obrázek 38 - Test validace HTML

4.4 Zhodnocení

Po implementaci nového rozšíření informačního systému byla vytvořena online schůzka, kde bylo rozšíření demonstrováno kolegům a vedoucímu oddělení. Veškeré inovace byly demonstrovány a jednotlivé pozice byly informovány o svých pravomocích a o tom, jak pracovat s vkládáním informací do informačního systému.

Na konci schůzky byly vyjádřeny pozitivní ohlasy s poděkováním od vedoucího a s doplňující otázkou, kdy by bylo možné informační systém začít využívat.

4.5 Výsledky a diskuse

Cílem diplomové práce bylo vytvořit informační systém v rozsahu, který byl požadován. Do informačního systému má vstup každý, kdo se do něj přihlásí. Toto je však povolené pouze kvůli prezentaci systému. V případě budoucí implementace by systém byl uzavřený a fungoval by se speciálními opatřeními na konkrétní síti.

Informační systém je responzivní a uživatel si ho může zobrazit na mobilních zařízeních. Při přechodu na mobilní zařízení se menu zabalí do „hamburgeru“.

Veškerá data, která jsou zobrazena v informačním systému, jsou fiktivní z důvodu ochrany osobních údajů. Jsou v systému uvedena z důvodu ukázky funkčnosti informačního systému.

Při rozšíření systému bylo již počítáno s maximální funkčností a efektivitou, avšak jako každý systém, i zde jsou místa pro vylepšení, pokud by o ně firma stála. Pokud bychom se měli zamyslet a navrhnout další rozšíření systému, pak by se systém mohl ještě rozšířit o další oblasti informačních a komunikačních systémů. Do webové stránky by se mohl také implementovat chat pro všechny v danou chvíli přihlášené uživatele. Kromě omezené správy uživatelů, kterou již informační systém obsahuje, by se mohla implementovat nová a propracovanější správa uživatelů a profilů v dané firmě.

5 Závěr

V teoretické části diplomové práce byly splněny veškeré dílčí cíle práce, které byly stanoveny. Dílčí cíle práce byly – zpřehlednění informačního systému a vytvoření modulu pro správu automobilů. Byly zde uvedeny možné procesy a metody, které se použily při vývoji tohoto informačního systému.

V praktické části diplomové práce jsme se věnovali analýze současného stavu, návrhu nového rozšíření informačního systému a jeho implementaci do stávajícího systému. Návrh byl podle úvodní analýzy zpracován pomocí UML a BPMN. UML nám zde nabídl různé diagramy – diagram tříd, diagram případů užití a stavový diagram.

Implementace byla provedena pomocí jazyka JavaScript. Bylo zde užito také databáze MySQL při ukládání dat o tankovacích kartách, výpůjčkách automobilů, událostech a správy automobilů. Aby byly tyto databáze využity, využil jsem také pro usnadnění práce phpMyAdmin.

Při návrhu systému byl kladen velký důraz na jednoduchost, jelikož účelem systému je zefektivnění a zrychlení práce uživatele (testera, analytika a dalších zainteresovaných osob), aby nemusel zdlouhavě vyhledávat veškeré potřebné informace.

Při tvorbě informačního systému jsem vycházel ze získaných znalostí ze všech předmětů, které jsem během studia absolvoval a ze zdrojů, které jsou uvedeny v seznamu použitých zdrojů a literatury.

6 Seznam použitých zdrojů

[1] CASTRO, Elizabeth a Bruce HYSLOP. *HTML5 a CSS3: názorný průvodce tvorbou WWW stránek*. Brno: Computer Press, 2012. ISBN 9788025137338.

[2] ČÁPKA, David. Lekce 2 - UML - Use Case Diagram. *ITnetwork* [online]. ©2020 [cit. 2020-03-28]. Dostupné z: https://www.itnetwork.cz/navrh/uml/uml-use-case-diagram

[3] MySQL, SQL a PHP. *Tvorba-Webu.cz* [online]. c2003-2008 [cit. 2020-02-15]. Dostupné z: https://www.tvorba-webu.cz/php/mysql.php

[4] Historical trends in the usage statistics of server-side programming languages for websites. *W3Techs* [online]. ©2009-2020 [cit. 2020-03-26]. Dostupné z: https://w3techs.com/technologies/history_overview/programming_language

[5] PROCHÁZKA, David. *CSS a XHTML: tvorba dokonalých WWW stránek krok za krokem*. 2., aktualiz. vyd. Praha: Grada, 2011. Průvodce (Grada). ISBN 9788024738970

[6] BAŠE, Ondřej. *JQuery pro neprogramátory: průvodce využitím knihovny jQuery UI*. Brno: Computer Press, 2012. ISBN 9788025137505

[7] BAAR, Ondřej. Historie SQL. *PCWorld* [online]. 2008 [cit. 2020-03-11]. Dostupné z: https://pcworld.cz/archiv/historie-sql-17391

[8] Začínáme s MySQL 1.díl. *Živě* [online]. Pavel Cvrček, 2001 [cit. 2020-02-15]. Dostupné z: https://www.zive.cz/clanky/zaciname-s-mysql-1dil/sc-3-a-102589/default.aspx

[9] Historie a vývoj HTML. *HTML5* [online]. Jirka Kosek, c1997-2014 [cit. 2020-03-05]. Dostupné z: http://htmlguru.cz/uvod-historie.html

[10] ŠINDELÁŘ, Jan. Historie webových prohlížečů, část 2. - WorldWideWeb a Mosaic. *Živě* [online]. 2005 [cit. 2020-03-06]. Dostupné z: https://www.zive.cz/clanky/historie-webovych-prohlizecu-cast-2---worldwideweb-a-mosaic/sc-3-a-127213/default.aspx

[11] SCHILDT, Herbert. *Java 8: výukový kurs*. Přeložil Jakub GONER. Brno: Computer Press, 2016. ISBN 9788025146651.

[12] ŽÁRA, Ondřej. *JavaScript: programátorské techniky a webové technologie*. Brno: Computer Press, 2015. ISBN 9788025145739.

[13] Úvod do JavaScriptu. *Jak psát web* [online]. Dušan Janovský [cit. 2020-03-01]. Dostupné z: https://www.jakpsatweb.cz/javascript/javascript-uvod.html

[14] SKLENÁK, Vilém. *Data, informace, znalosti a Internet*. Praha: C.H. Beck, 2001. C.H. Beck pro praxi. ISBN 80-7179-409-0.

[15] TVRDÍKOVÁ, Milena. *Aplikace moderních informačních technologií v řízení firmy: nástroje ke zvyšování kvality informačních systémů*. Praha: Grada, 2008. Management v informační společnosti. ISBN 9788024727288.

[16] ARLOW, Jim a Ila NEUSTADT. *UML 2 a unifikovaný proces vývoje aplikací: objektově orientovaná analýza a návrh prakticky*. 2., aktualiz. a dopl. vyd. Brno: Computer Press, 2007. ISBN 978-80-251-1503-9.

[17] HLAVA, Tomáš. Vodopádový model. *Testování softwaru* [online]. 2016 [cit. 2020-03-21]. Dostupné z: http://testovanisoftwaru.cz/manualni-testovani/modely-zivotniho-cyklu-softwaru/vodopadovy-model/

[18] Vodopádový model (Waterfall model). *Management Mania* [online]. 2015 [cit. 2020-03-05]. Dostupné z: https://managementmania.com/cs/vodopadovy-model-waterfall-model

[19] Vývojové modely. Diagnostika a testování elektronických systémů [online]. ©2012 [cit. 2020-01-22]. Dostupné z: http://www.umel.feec.vutbr.cz/bdts/index.php/embedded-systemy/vyvojove-modely

[20] BRUCKNER, Tomáš. *Tvorba informačních systémů: principy, metodiky, architektury*. Praha: Grada, 2012. Management v informační společnosti. ISBN 978-80-247-4153-6.

[21] CHLEBOVSKÝ, Vít. *Management zákaznických řešení: jak efektivně tvořit a spravovat individualizovaná řešení zákaznických potřeb*. První vydání. Praha: Grada, 2017. Manažer. ISBN 9788027105595.

[22] Agilní metodiky řízení vývoje software (Agile software development methodologies). *Management Mania* [online]. 2016 [cit. 2020-03-05]. Dostupné z: https://managementmania.com/cs/agilni-metodiky-rizeni-vyvoje-software

[23] BECK, Kent. *Extrémní programování*. Praha: Grada, 2002. ISBN 9788024703008.

[24] KOLESNIKOV, Pavel. Extrémní programování v praxi. *Interval* [online]. 2002, [cit. 2020-03-11]. Dostupné z: https://www.interval.cz/clanky/extremni-programovani-v-praxi/

[25] BECK, Kent. *Programování řízené testy*. Praha: Grada, 2004. Moderní programování. ISBN 8024709015

[26] KNESL, Jiří. Co je Scrum? Jiří Knesl [online]. 2016 [cit. 2020-03-08]. Dostupné z: http://www.knesl.com/co-je-scrum

[27] ISO/IEC 19510:2013. *ISO* [online]. 2013 [cit. 2020-03-31]. Dostupné z: https://www.iso.org/standard/62652.html

[28] SODOMKA, Petr a Hana KLČOVÁ. *Informační systémy v podnikové praxi*. 2., aktualiz. a rozš. vyd. Brno: Computer Press, 2010. ISBN 978-80-251-2878-7.

[29] ČÁPKA, David. Lekce 1 - Úvod do UML. *ITnetwork* [online]. ©2020 [cit. 2020-02-11]. Dostupné z: https://www.itnetwork.cz/navrh/uml/uml-uvod-historie-vyznam-a-diagramy/

[30] ČÁPKA, David. Lekce 2 - UML - Use Case Diagram. *ITnetwork* [online]. ©2020 [cit. 2020-02-11]. Dostupné z: https://www.itnetwork.cz/navrh/uml/uml-use-case-diagram

[31] Pavus. Use case diagram - diagram případů užití. *Mpavus* [online]. 2006 [cit. 2020-02-22]. Dostupné z: http://mpavus.wz.cz/uml/uml-b-use-case-3-2-1.php

[32] ČÁPKA, David. Lekce 5 - UML - Class diagram. *ITnetwork* [online]. ©2020 [cit. 2020-03-11]. Dostupné z: https://www.itnetwork.cz/navrh/uml/uml-class-diagram-tridni-model

[33] Pavus. Class diagram - diagram tříd. Mpavus [online]. 2006 [cit. 2020-03-11]. Dostupné z: http://mpavus.wz.cz/uml/uml-s-class-3-3-1.php

[34] PASTORČÁK, Petr. Diagram tříd (class diagram). *Objektově orientované modelování systémů* [online]. ©2004 [cit. 2020-04-05]. Dostupné z: http://www.orca.xf.cz/ooms/001/001.htm

[35] ČÁPKA, David. Lekce 4 - UML - Doménový model. *ITnetwork* [online]. ©2020 [cit. 2020-04-05]. Dostupné z: https://www.itnetwork.cz/navrh/uml/uml-domenovy-model-diagram

7 Přílohy

Na přiloženém disku se v kořenovém adresáři nachází tato diplomová práce ve formátu zaverecna_prace.pdf a složka Manažerský informační systém správy automobilů, ve které jsou soubory s celým informačním systém.