

MASARYKOVA UNIVERZITA  
FAKULTA INFORMATIKY



# **Návrh a implementace informačního systému pro stravovací zařízení**

BAKALÁRSKA PRÁCA

**Jakub Adamus**

Brno, 2013

## **Prehlásenie**

Prehlasujem, že táto bakalárska práca je mojím pôvodným autorským dielom, ktoré som vypracoval samostatne. Všetky zdroje, pramene a literatúru, ktoré som pri vypracovaní používal alebo z nich čerpal, v práci riadne citujem s uvedením úplného odkazu na príslušný zdroj.

Jakub Adamus

**Vedúci práce:** RNDr. Jaroslav Škrabálek, MBA

## **Pod'akovanie**

Chcem sa pod'akovať môjmu vedúcemu bakalárskej práce RNDr. Jaroslavu Škrabálkovi za cenné rady, námety, ale hlavne za čas, ktorý mi venoval.

## Zhrnutie

Práca sa zaoberá vytvorením informačného systému pre stravovacie zariadenie. V úvodnej časti práce sa venujem popisu stravovacích zariadení. V ďalších kapitolách opisujem celkový proces vytvorenia informačného systému. Na začiatku procesu identifikujem a následne overím problémy, ktoré majú používatelia stravovacích systémov. Ďalej sa venujem opisu životného cyklu informačného systému od analýzy a návrhu, cez implementáciu až po testovanie. Informačný systém má názov **Obedár**.

## **Klíčové slová**

informačný systém, stravovací softvér, web, RFID, softvérový proces, záverečná práca, ASP.NET, LINQ, SQL, web forms, membership, lean canvas, lean

## Obsah

|       |                                                              |    |
|-------|--------------------------------------------------------------|----|
| 1     | Úvod . . . . .                                               | 3  |
| 2     | Stravovacie zariadenia . . . . .                             | 4  |
| 2.1   | Zamestnanecké stravovacie zariadenia . . . . .               | 4  |
| 2.1.1 | Procesy v zamestnaneckých stravovacích zariadeniach          | 5  |
|       | Tvorba jedálneho lístka . . . . .                            | 5  |
|       | Objednanie . . . . .                                         | 5  |
|       | Príjem jedál/surovín . . . . .                               | 5  |
|       | Rozdelenie na porcie . . . . .                               | 6  |
|       | Overenie stravníka . . . . .                                 | 6  |
|       | Výdaj jedla . . . . .                                        | 6  |
|       | Upratovanie . . . . .                                        | 6  |
| 2.1.2 | IS pre zamestnanecké stravovacie zariadenia . . . . .        | 6  |
| 3     | Počítačový informačný systém . . . . .                       | 8  |
| 3.1   | Nevýhody . . . . .                                           | 8  |
| 3.1.1 | Bezpečnosť . . . . .                                         | 8  |
| 3.1.2 | Objednanie . . . . .                                         | 8  |
| 3.1.3 | Jedálny lístok . . . . .                                     | 9  |
| 3.1.4 | Overenie stravníka . . . . .                                 | 9  |
| 3.1.5 | Reklamácia objednávok . . . . .                              | 9  |
| 3.1.6 | Sumarizácia obedov . . . . .                                 | 9  |
| 4     | Prvky lean a validácia . . . . .                             | 10 |
| 4.1   | Lean Canvas . . . . .                                        | 10 |
| 4.1.1 | Customer segments . . . . .                                  | 11 |
|       | Stravník . . . . .                                           | 11 |
| 4.1.2 | Problem . . . . .                                            | 11 |
| 4.1.3 | Solution . . . . .                                           | 12 |
| 4.1.4 | Unfair advantage . . . . .                                   | 12 |
| 4.1.5 | Revenue streams . . . . .                                    | 12 |
| 4.1.6 | Cost structure . . . . .                                     | 13 |
| 4.2   | Validácia . . . . .                                          | 13 |
| 4.2.1 | Prieskum . . . . .                                           | 13 |
|       | 1. Vlastníte <i>smartphone</i> ? . . . . .                   | 13 |
|       | 2. Ako často objednávate? . . . . .                          | 14 |
|       | 3. Je objednávanie na PC pre Vás nepraktické? . . . . .      | 15 |
|       | 4. Máte záujem o históriu svojich objednávok? . . . . .      | 15 |
|       | 5. Využili by ste objednávanie jedál cez internet? . . . . . | 16 |
| 5     | Analýza IS . . . . .                                         | 17 |

---

|       |                                         |    |
|-------|-----------------------------------------|----|
| 5.1   | <i>Model softvérového procesu</i>       | 17 |
| 5.2   | <i>Analýza</i>                          | 18 |
| 5.2.1 | <i>Stakeholders</i>                     | 18 |
|       | Funkčné požiadavky                      | 18 |
|       | Nefunkčné požiadavky                    | 20 |
| 6     | <b>Návrh IS</b>                         | 21 |
| 6.1   | <i>Diagram tried</i>                    | 21 |
| 6.2   | <i>Databáza</i>                         | 23 |
| 6.2.1 | ERD (Entity–Relationship diagram)       | 23 |
|       | Entita                                  | 23 |
|       | Entitná množina                         | 23 |
|       | Relácia                                 | 23 |
|       | Početnosť relácii                       | 23 |
| 7     | <b>Implementácia</b>                    | 25 |
| 7.1   | <i>Použité technológie</i>              | 25 |
| 7.1.1 | ASP.NET                                 | 25 |
|       | MVC                                     | 26 |
|       | Web Forms                               | 26 |
| 7.1.2 | Entity Framework                        | 26 |
|       | Kontext databáze                        | 27 |
| 7.1.3 | LINQ                                    | 28 |
|       | LINQ to Entities                        | 28 |
| 7.1.4 | Responsive web design                   | 29 |
| 7.2   | <i>Informačný systém Obedár</i>         | 29 |
| 7.2.1 | Štruktúra                               | 29 |
|       | Web.config                              | 29 |
|       | Webstránky                              | 31 |
| 7.2.2 | Bezpečnosť                              | 31 |
|       | Používateľské role                      | 31 |
|       | Autorizácia                             | 31 |
|       | Prihlásenie                             | 32 |
| 7.3   | <i>Obedar.Scheduler</i>                 | 32 |
| 8     | <b>Testovanie</b>                       | 33 |
| 8.1   | <i>Unit Test</i>                        | 33 |
| 8.1.1 | NUnit framework                         | 33 |
| 9     | <b>Záver</b>                            | 34 |
| A     | <b>Obsah CD</b>                         | 37 |
| B     | <b>Lean Canvas Model</b>                | 38 |
| C     | <b>Ukážky používateľského rozhrania</b> | 39 |

# 1 Úvod

Informačný systém (IS) je softvérové vybavenie firmy, ktoré je schopné na základe spracovávaných informácií riadiť procesy podniku, alebo poskytovať tieto informácie riadiacim pracovníkom tak, aby boli schopní vykonávať riadiace funkcie, medzi ktoré patrí plánovanie, koordinácia a kontrola všetkých procesov firmy.[12]

Rýchlosť starnutia produktov v informačných technológiách je asi najrýchlejší proces starnutia dnešnej doby. Informačné systémy sa stali bežnou súčasťou našich životov.

Pohodlnosť používateľov prinútila mnoho firiem k neustálym zmenám. Celosvetový trend určujú webové informačné systémy. Mnoho spoločností má záujem používať webové informačné systémy, ktoré umožnia prístup k dátam z akéhokoľvek miesta s prístupom na internet. Interné systémy pre manažment a prideľovanie úloh zamestnancom sú dávno dostupné z webového rozhrania. Výhody tkvejú v organizovaní procesov z pohodlia domova alebo na cestách.

Ďalšie výhody sú pochopiteľne v prenositeľnosti a v kompatibilite, pretože prihlásiť sa do systému, ktorý zobrazuje klasické *HTML* webstránky je pre všetky komunikačné zariadenia ako sú mobilné telefóny, tablety, počítače a dokonca už aj televízory maličkosťou. *Desktopové* aplikácie sa pomaly stávajú históriou. Nevyhnutný je len internetový prehliadač, pričom nezáleží na operačnom systéme zariadenia.

Táto práca sa zaoberá vytvorením webového informačného systému pre stravovacie zariadenie. Systém slúži hlavne spoločnosti **DOR s.r.o.** so sídlom v Považskej Bystrici na Slovensku. Na začiatku práce opisujem stravovacie zariadenia vo všeobecnosti. Pomocou systémovej analýzy určím hlavné procesy, ktoré stravovacie zariadenia vykonávajú. V ďalšej časti sa venujem hlavne prvkom metodiky *lean*, s ktorou súvisí prieskum uskuutočnený medzi zamestnancami. Následne popíšem životný cyklus tvorby stravovacieho informačného systému, ktorý je výsledkom tejto práce. Systém je vytvorený pomocou technológie **.NET**, presnejšie webový *framework* **ASP.NET** verzie 4.5.



## 2 Stravovacie zariadenia

V tejto kapitole opisujem stravovacie zariadenia. Procesy, ktoré sú nevyhnutné na správne fungovanie týchto zariadení.

Príčinou vzniku zamestnaneckého stravovania je rýchly rozvoj priemyselnej veľkovýroby. Hlavným dôvodom bolo docieľiť vysokú produktivitu práce s minimom časových strát. Význam zamestnaneckého stravovania [5, s. 27]:

- umožňuje stravovanie zamestnancov v priebehu pracovného procesu k uspokojeniu základných potrieb výživy
- poskytuje diferencované stravovanie v závislosti na energetickom výdaji počas pracovného procesu
- zabezpečuje aj doplnkové stravovanie zamestnancov zaistením možnosti občerstvenia

Stravovanie pracovníkov v priebehu pracovného procesu je dôležité z hľadiska udržania pracovného výkonu a zaistenia plynulého prísunu energetických a nutričných látok potrebných pre ľudský organizmus. Prispieva k spokojnosti zamestnancov a príjemnejšej atmosfére na pracoviskách.

Existujú rôzne formy stravovacích zariadení. Ja opisujem len zamestnanecké stravovacie zariadenia, pre ktoré je určený nový informačný systém.

### 2.1 Zamestnanecké stravovacie zariadenia

Zamestnanecké stravovacie zariadenia sú zariadenia, ktoré poskytujú stravovacie služby zamestnancom. Stravníci si neobjednávajú jedlo u čašníka, ale fungujú na princípe samoobsluhy. Podľa počtu pracovníkov sa rozdeľujú na [5, s. 17]:

- malé (0–19 zamestnancov)
- stredné (20–100 zamestnancov)
- veľké (nad 100 zamestnancov)

Každé stravovacie zariadenie potrebuje nejakú stratégiu na prijímanie objednávok, poskytovanie informácii o jedálnych lístkoch. Ešte stále existuje mnoho firiem, prípadne škôl, ktoré využívajú klasickú papierovú evidenciu k organizácii stravovacieho zariadenia. Avšak, papierová evidencia poskytuje veľmi obmedzené možnosti kontroly a riadenia prevádzok. Časovo

náročný proces vytvárania stravných lístkov, papierovej evidencie zd'aleka neposkytuje dostatočné množstvo a štruktúru informácii, ktoré sú k riadeniu a kontrole nevyhnutné. Pri tomto spôsobe evidencie veľmi záleží na dôveryhodnosti personálu, pretože papierová evidencia sa môže veľmi ľahko vedome sfalšovať.

### 2.1.1 Procesy v zamestnaneckých stravovacích zariadeniach

Všetky spoločnosti majú stanovené určité procesy, ktoré slúžia hlavne k správaniu porozumeniu povinností zamestnancov. V praxi je úplne bežné, že dve rôzne spoločnosti, ktoré majú rovnaký podnikateľský zámer, môžu mať procesy diametrálne odlišné.

V tejto sekcii opisujem hlavné procesy, ktoré sú vykonávané v zamestnaneckých stravovacích zariadeniach vo všeobecnosti. Pri každom procese opisujem, aká aktivita proces zaháji, a čo je výsledkom procesu. Tieto procesy mi umožnili definovať požiadavky na systém, ktorým sa venujem v analýze IS (pozri 5.2).

#### Tvorba jedálneho lístka

Proces v sebe zahŕňa tvorbu jedálneho lístka na určité časové obdobie. Najčastejšie sa vytvára týždeň vopred. Tento proces sa vykonáva v stravovacích zariadeniach, ktoré jedlá aj pripravujú. V opačnom prípade je tento proces uskutočňovaný dodávateľskou spoločnosťou.

**Začiatok procesu:** Nastane koniec časového obdobia, napríklad týždňa.

**Výsledok procesu:** Výsledkom je jedálny lístok, ktorý obsahuje jedlá rozvrhnuté na určité časové obdobie.

#### Objednanie

Každý stravník musí vytvoriť objednávku. Pričom nie je dôležité, či sa pod pojmom objednanie rozumie vytvorenie objednávky pomocou nejakého softvéru, alebo len jednoduchým vložením lístka s identifikačnými údajmi do papierovej schránky.

**Začiatok procesu:** Stravník si objedná jedlo.

**Výsledok procesu:** Spracovanie objednávky.

#### Príjem jedál/surovín

Príjem jedál prípadne surovín, z ktorých sa jedlá vyhotovujú.

**Začiatok procesu:** Dodanie objednávok.

**Výsledok procesu:** Jedlá sú pripravené na rozdelenie na porcie.

Rozdelenie na porcie

Predstavuje rozdelenie jednotlivých jedál na porcie.

**Začiatok procesu:** Príjem hotových jedál.

**Výsledok procesu:** Porcia pripravená na výdaj pre stravníka.

Overenie stravníka

Každý stravník sa musí preukázať, či má nárok na požadovanú porciu jedla. Tento proces môže mať opäť viacero foriem. Najdôveryhodnejší spôsob je však využitie nejakého softvéru a identifikačnej kartičky, ktorú má každý stravník.

**Začiatok procesu:** Požiadavka na výdaj jedla

**Výsledok procesu:** Stravník je overený.

Výdaj jedla

Proces kedy kuchárka odovzdá porciu jedla stravníkovi.

**Začiatok procesu:** Identifikácia stravníka.

**Výsledok procesu:** Stravník dostane porciu a objednávka je následne spracovaná.

Upratovanie

Nevyhnutná súčasť každého stravovacieho zariadenia. Zahrňuje v sebe ďalšie procesy, akými sú: umývanie riadu a kuchynského náradia, príprava stolov atď.

**Začiatok procesu:** Koniec výdaja jedál.

**Výsledok procesu:** Jedáleň je opätovne pripravená na stravovanie.

### 2.1.2 IS pre zamestnanecké stravovacie zariadenia

Informačné systémy sú nevyhnutnou súčasťou našich životov. Tieto systémy dokážu uľahčiť prácu a zaviesť skutočnú organizáciu tam, kde predtým bola len zdanlivo. Pre efektívnu evidenciu a kontrolu je nevyhnutné zaviesť informačný systém.

Informačný systém pre stravovacie zariadenie je základný systém, ktorý ukladá a zaznamenáva stravníkov, ktorí využívajú služby v stravovacom

zariadení. Informačný systém musí byť navrhnutý tak, aby spĺňal požiadavky zainteresovaných ľudí. Nasledujúci zoznam obsahuje hlavné potreby na informačný systém pre stravovacie zariadenia:

- identifikácia stravníkov
- prijímanie objednávok od stravníkov
- sumarizácia objednávok
- história objednávok pre každého zamestnanca
- plánovanie jedálnych lístkov
- finančná kontrola
- skoncovanie s papierovými stravnými lístkami
- kontrola prevádzky

V systéme je vždy overená totožnosť stravníka nejakým identifikačným zariadením, typicky čipovou kartou. Existujú však samoobslužné stravovacie zariadenia, ktoré sú určené širokej verejnosti a nielen zamestnancom. Vtedy plynie nová požiadavka na stravovací systém – prepojenie na pokladničný systém z dôvodu hotovostných platieb prijatých od zákazníkov.

V nasledujúcej kapitole opisujem počiatočný informačný systém, ktorý spoločnosť používala.

### 3 Počiatočný informačný systém

V tejto kapitole popisujem počiatočný systém v spoločnosti **DOR s.r.o.** Firma má v prevádzke stravovací informačný systém s názvom **ObeDOR**. Skladá sa z troch rôznych spustiteľných súborov, ktoré komunikujú s databázovým serverom:

1. **ObeDOR\_Kucharka.exe** – je spustený na počítači u kuchárky. Pri prihlásení stravníka do systému zobrazí jeho objednávky na aktuálny deň.
2. **ObeDOR\_Objednavky.exe** – beží na počítači v kantíne. Slúži na objednávanie, prípadne rušenie objednávok.
3. **ObeDOR\_Ekonomka.exe** – zobrazí sumarizáciu obedov, umožňuje rušiť staršie objednávky.

#### 3.1 Nevýhody

Táto sekcia opisuje niektoré procesy, v ktorých som identifikoval veľmi závažné problémy.

##### 3.1.1 Bezpečnosť

Súbory musia byť chránené rôznymi oprávneniami systému, aby sa zamestnanec nemohol k spustiteľným súborom dostať. Sú v nich totiž uložené prihlasovacie údaje na databázu. Programy sú spustiteľné bez zadania hesla, avšak fungujú len na lokálnej sieti v spoločnosti.

##### 3.1.2 Objednanie

Proces vytvorenia objednávky je v systéme **ObeDOR** veľmi zdĺhavý. Skladá sa z viacerých činností:

1. Fyzický príchod stravníka k počítaču umiestneného v kantíne.
2. Autentifikácia stravníka.
3. Výber dátumu pomocou počítačovej myši.
4. Výber jedla z jedálneho lístku.
5. Odhlásenie zo systému.

#### 3.1.3 Jedálny lístok

V informačnom systéme jedálny lístok nie je implementovaný. Stravník má na výber vždy z 10 obedov, ktoré sú v systéme evidované ako obed č.1, č.2, ..., č.10. Tieto možnosti sú uložené priamo v spustiteľnom programe. Externá firma dodáva jedlá a jedálne lístky. Dodávateľ neposkytuje vždy až 10 jedál na každý deň. Bežne vznikajú reklamácie na objednávky, ktoré boli vytvorené nepozornosťou stravníkov.

#### 3.1.4 Overenie stravníka

Overenie prebieha pomocou čítacieho zariadenia, ktoré sníma čiarový kód, konkrétne *International Article Number* (EAN-13). Každý stravník disponuje s kartičkou, na ktorej má vytlačený tento identifikátor.

#### 3.1.5 Reklamácia objednávok

Ekonom musí mať privilégium vymazávať objednávky kvôli riziku vytvorenia neplatných objednávok. V takomto prípade veľmi záleží na dôveryhodnosti ekonomického oddelenia.

#### 3.1.6 Sumarizácia obedov

Systém dokáže vygenerovať sumarizáciu obedov, ktoré si každý stravník za určité časové obdobie objednal. Problém však je, že systém podporuje len jednotnú cenu obedov.

Tvorba informačného systému v sebe zahŕňa rôzne aktivity, ktoré umožnia uľahčiť prácu vývojárom. Je veľmi dôležité vytvárať systém postupnými krokmi. Tieto kroky slúžia aj ako dokumentácia, ktorá zjednodušuje prácu a šetrí čas pri ďalšom vývoji. V nasledujúcej kapitole sa venujem modelu *lean canvas*, vďaka ktorému som získal podrobný prehľad, aké funkcie má nový informačný systém pre spoločnosť **DOR s.r.o.** obsahovať.

## 4 Prvky lean a validácia

V tejto kapitole veľmi stručne opíšem metodiku *lean*. Následne opisujem môj *lean canvas* model. Ďalšiu časť tvorí prieskum, vďaka ktorému som mohol vyvodiť rôzne dôsledky pri tvorbe informačného systému **Obedár**.

*Lean manufacturing* – štíhla výroba. Často sa používa len krátky výraz *lean*. *Lean* je metodika výroby, ktorú vyvinula spoločnosť **Toyota**. Filozofia spočíva v minimalizovaní plytvania zdrojmi. Cieľom je uspokojovať potreby zákazníka a vytvárať pre neho nejaké hodnoty. Plytvanie zdrojmi sa chápe ako vykonávanie akejkoľvek aktivity, ktorá má odlišný cieľ.

Z metodiky *lean* bol odvodený tzv. *lean startup*, ktorý sa používa najmä v aplikovanej informatike. *Lean startup* je metodika zahájenia produktu, ktorej hlavná myšlienka spočíva vo validácii problémov. Je dôležité vyvíjať systém tak, aby sa efektívne využíval čas a neriešili sa problémy, ktoré v skutočnosti ani nie je nutné vyriešiť. Tento spôsob automaticky vedie ku skráteniu vývojového cyklu a teda k zamedzeniu plytvania zdrojov.

Na aplikovanie *lean startup* metodiky som použil model s názvom *lean canvas*, ktorý opisujem v nasledujúcej sekcii. Aplikovanie metodiky uľahčuje prácu analytikom, vývojárom ale aj projektovému manažérovi IS.

### 4.1 Lean Canvas

Používa sa na tvorbu jednostránkových biznis modelov. Vytvoriť klasický biznis plán môže zabráť niekoľko týždňov až mesiacov. *Lean Canvas* umožňuje zachytiť viacero biznis modelov vo veľmi krátkom časovom rozmedzí – napríklad za jedno popoludnie. Využíva *brainstorming*, pričom je dôležité vyberať správne slová a správne vety. Autor sa musí vtesnať s vytvorením produktu na jednu stránku.

Základným stavebným kameňom je získavanie informácií od zákazníkov. Čo umožňuje určiť, či má zákazník skutočne o riešenie jeho problémov záujem.

Je rozdelený na viacero častí: [2, s. 27]

1. *Customer segments* – zákazníci
2. *Problem* – 1–3 problémy
3. *Unique value proposition* – presvedčivá správa
4. *Solution* – riešenie (problémov)

5. *Unfair advantage* – výhody
6. *Revenue streams* – zdroj príjmov
7. *Cost structure* – fixné a variabilné náklady
8. *Key metrics* – metriky odzrkadľujúce vývoj biznisu
9. *Channels* – cesta k zákazníkom

Táto sekcia obsahuje popis niektorých častí *lean canvas* modelu, ktorý som vytváral počas vývoja. Celý model sa nachádza v prílohe B.

#### 4.1.1 Customer segments

Do tejto skupiny patria zákazníci a používatelia. Zákazníci sú tí, ktorí sú ochotní zaplatiť za produkt. Používatelia používajú produkt.

Zákazníci pre stravovací systém **Obedár** predstavujú akékoľvek organizácie, ktoré majú jedáleň. Typicky to môžu byť firmy so zamestnaneckým stravovaním, prípadne školy so študentmi. V mojom prípade je prvým zákazníkom spoločnosť **DOR s.r.o.**, pričom stravníci v tejto spoločnosti predstavujú používateľov.

##### Stravník

Stravník je osoba, ktorá môže využívať stravovacie služby v organizácii. Môže to byť každý zamestnanec, prípadne sa môže jednať o žiakov, resp. študentov. Stravníci sú v tomto prípade zároveň používatelia systému **Obedár**. Na používateľov som sa v mojom prípade zameral. Sú to ľudia, ktorí mi môžu poskytnúť *feedback* (spätnú väzbu).

#### 4.1.2 Problem

Táto časť slúži na definovanie jedného až troch hlavných problémov, ktoré by mal informačný systém **Obedár** vyriešiť. Identifikácia problémov je nevyhnutnou časťou vývoja úspešného produktu.

1. **fyzická prítomnosť na vytváranie objednávok** – používateľ musí byť fyzicky prítomný v kantíne, aby si mohol objednať obedy na nasledujúce dni.
2. **rad v jedálni** – v dôsledku nutnej fyzickej prítomnosti pri objednávaní vzniká v jedálni rad čakajúcich ľudí.



3. **kontrola objednávok** – stravník nemá prístup k histórii objednávok. Pokiaľ teda má záujem o overenie platieb za objednané jedlá, je nútený robiť si vlastnú evidenciu.

#### 4.1.3 Solution

Zákazníkov trápia problémy. Úlohou vývojára je tieto problémy vyriešiť.

**Solution** – Riešenie.

Navrhnuté riešenie sa skladá z týchto dvoch bodov:

- webové rozhranie
- história objednávok

Webové rozhranie považujem za najlepšie vyriešenie problémov, ktoré trápia stravníkov. Používateľ bude automaticky schopný zobrazíť históriu objednávok cez internetový prehliadač z pohodlia domova. Bude si môcť objednať jedlo z domu. Fronty v jedálni kvôli jednému počítaču automaticky zmiznú.

#### 4.1.4 Unfair advantage

Predstavujú výhody produktu. Do tejto časti patrí história objednávok a jej predikcia k zníženiu nákladov.

#### 4.1.5 Revenue streams

Zdroje príjmov na pokračovanie vývoja systému. **Prvotná inštalácia a zavedenie systému** sa bude pohybovať len okolo 199 €. Zahŕňa v sebe nasledujúce služby:

1. konfigurácia IIS (*Internet Information Services* – serverová aplikácia vyvíjaná firmou Microsoft)
2. konfigurácia *Microsoft SQL Server*
3. nastavenie systému **Obedár** – konfigurácia SMTP (*Simple Mail Transfer Protocol* – zabezpečuje odosielanie emailov).

**Zaškolenie zamestnancov** je samostatná služba. Jedná sa o 3 hodinovú prednášku s následnou demonštráciou. Cieľom je, aby zamestnanci porozumeli ovládaniu IS.

Pravidelný príjem 99 € za rok slúži na nový vývoj a aktualizácie.

Pri poruchách, ktoré neboli zapríčinené chybou dodávaného softvéru je servis vykonávaný odborným technikom ohodnotený na 24,99 €/hod.

#### 4.1.6 Cost structure

##### Náklady

- mesiac intenzívneho vývoja produktu ~1500 € – zahŕňa v sebe náklady na vývoj systému, čas, elektrickú energiu atď’.
- stretnutia so zainteresovanými ľuďmi v spoločnosti (pozri 5.2.1) ~100 € – náklady len na dopravu
- softvér ~615 € (Visual Studio Professional 2012)

#### 4.2 Validácia

Takmer každý produkt, pri ktorom sa nevykoná validácia problémov je odsúdený na neúspech. Vytvoriť aplikáciu, ktorá rieši neexistujúci problém, respektíve rieši problém, s ktorým dokáže každý žiť je odkázaná na neúspech. Preto bolo pre mňa nevyhnutné overiť, či zákazník a používatelia skutočne potrebujú vyriešiť problémy, ktoré majú.

Do prieskumu boli zapojení stravníci, kuchárky ktoré majú na starosti vydávať stravníkom jedlo, ľudia z ekonomického oddelenia ale aj manažment a riaditeľ. Formulár vyplnilo presne 137 ľudí.

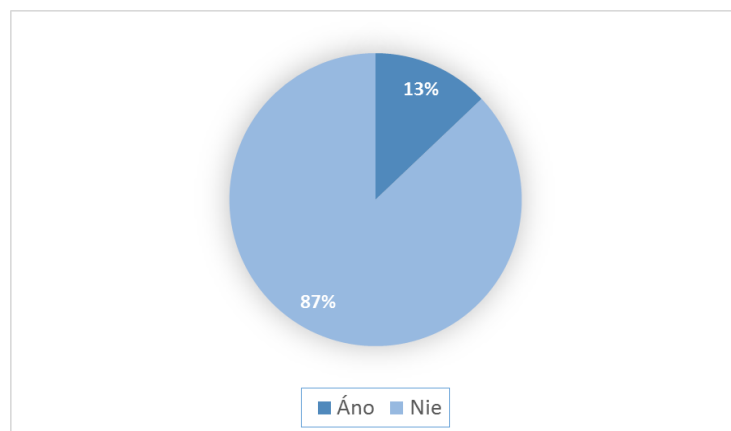
##### 4.2.1 Prieskum

Cieľom prieskumu má byť potvrdenie, či problémy, ktoré som sa rozhodol riešiť sú skutočne problémy, ktoré musia byť vyriešené a teda, či má zmysel kvôli nim míňať zdroje a čas.

##### 1. Vlastníte *smartphone*?

*Smartphone* – mobilný telefón, ktorý má nainštalovaný jeden z operačných systémov: *Windows Phone*, *BlackBerry*, *Android*, *iOS (Apple)*.

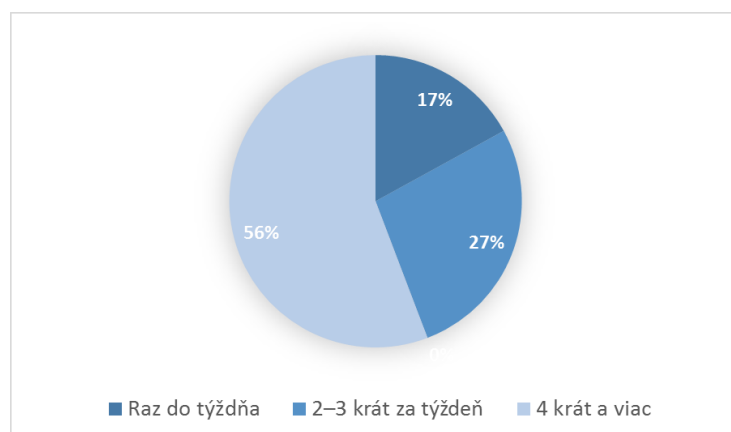
Až 87 % dotazovaných zvolilo odpoveď nie (obrázok 4.1). Vďaka tejto informácii som považoval vývoj klienta na rôzne mobilné platformy za zbytočné plytvanie zdrojmi. Priorita sa automaticky znížila, avšak rozhodol som sa vytvoriť aspoň prívetivejšie používateľské rozhranie s využitím tzv. „responzívneho“ webového dizajnu (pozri 7.1.4).



Obr. 4.1: Výsledný graf 1. otázky

## 2. Ako často objednávate?

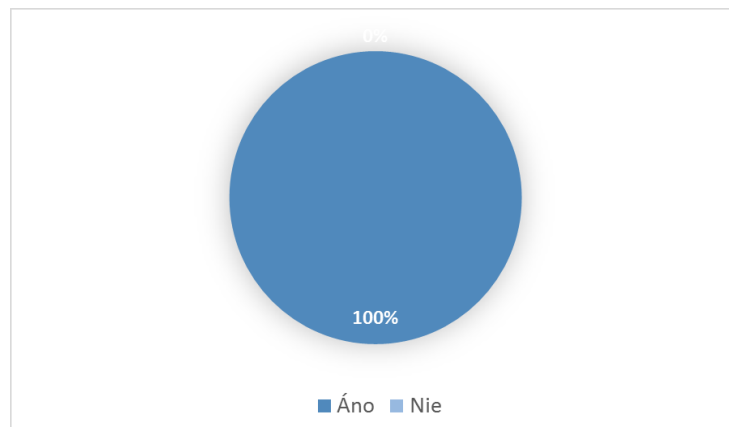
Z grafu (obrázok 4.2) vyplýva, že až 56 % stravníkov musí prísť do jedálne k počítaču viac ako štyrikrát do týždňa za účelom objednávania (pozri. 3.1.2). Priorita riešiť tento problém sa teda zvýšila.



Obr. 4.2: Výsledný graf 2. otázky

3. Je objednávanie na PC pre Vás nepraktické?

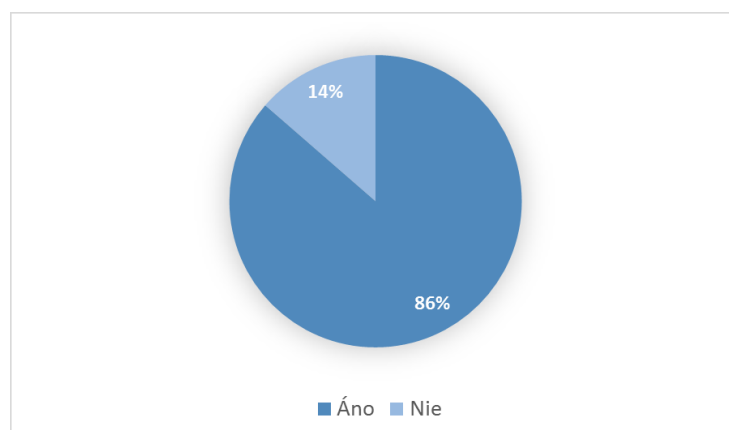
Otázka č. 3 (obrázok 4.3) slúžila k prevereniu, či problém zdĺhavého objednávanie je skutočne problém, ktorý je nevyhnutné vyriešiť (pozri. 3.1.2).



Obr. 4.3: Výsledný graf 3. otázky

4. Máte záujem o históriu svojich objednávok?

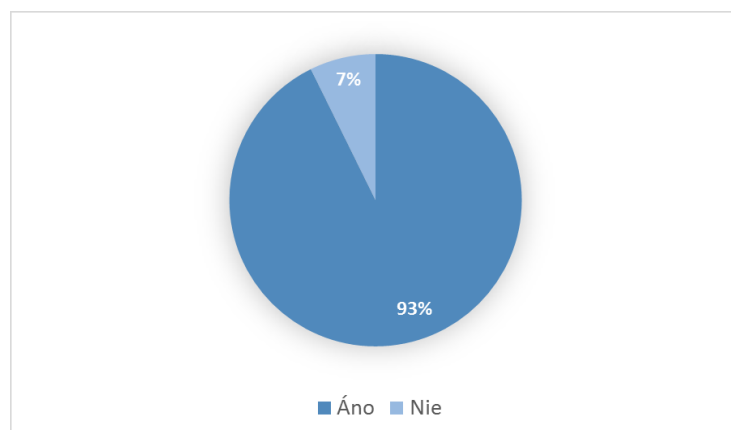
Až 86 % respondentov (obrázok 4.4) zdvihlo prioritu problému s históriou na vysokú úroveň. Preto bude musieť byť história bezpochyby implementovaná.



Obr. 4.4: Výsledný graf 4. otázky

## 5. Využili by ste objednávanie jedál cez internet?

Len 7 % z dotazovaných (obrázok 4.5) vybralo zápornú odpoveď. Táto otázka teda s istotou potvrdila, že je skutočne nevyhnutné vytvoriť systém tak, aby stravník mohol objednávať bez fyzickej prítomnosti v jedálni. Tým sa vyrieši aj jednoduchý prístup k histórii objednávok.



Obr. 4.5: Výsledný graf 5. otázky

**Zhodnotenie:** Vývojári majú tendenciu vytvárať rôznu funkcionality v systémoch v domnienke, že je o ňu vždy záujem. Prvky *lean* mi umožnili správne zhodnotiť problémy a či má skutočne význam vytvárať rôznu funkcionality, alebo je to v aktuálnom čase len plytvanie zdrojmi. V nasledujúcej kapitole opisujem analýzu informačného systému, kde využijem poznatky získané vďaka *lean*.

## 5 Analýza IS

Softvérový proces je štruktúrovaná množina aktivít, potrebná na vytvorenie softvéru. Táto množina v sebe môže obsahovať aj aktivity, ktoré súvisia s úplným začiatkom tvorby softvéru. Vytvorenie softvéru od nuly je však pre všetky spoločnosti príliš nákladný proces. Aj preto sa firmy upierajú smerom využívania už existujúcich častí svojho softvéru, prípadne tretích strán. Týmto spôsobom spoločnosti ušetria roky vývoja svojich zamestnancov.

### 5.1 Model softvérového procesu

Model softvérového procesu je model, ktorý umožňuje zachytiť zjednodušený softvérový proces. Väčšina modelov je založená na jednom z nasledujúcich generických vzorov: [10, s. 9]

- **Vodopádový model** – Jeden z najstarších a najznámejších modelov. Vývoj softvéru je rozdelený na viacero častí pričom sa sekvenčne postupuje z jednej časti do druhej. Rozdelenie na časti má za následok príliš zložité úpravy pri zmenených požiadavkách klientov. Vodopádový model je teda vhodný pre firmy, ktoré majú presne definované a nemenné požiadavky, tým pádom dokážu presne určiť špecifikáciu softvéru. Používa sa hlavne pri krabicových softvéroch.
- **Inkrementálny vývoj** – Pracuje sa s tzv. inkrementmi. Inkreментy vždy prinesú nejakú zmenu, ktorá je vyhotovená na základe komentárov a spätnej väzby od používateľov softvéru. Cena pri zmene požiadavkov je redukovaná. Štruktúra systému má pri veľkých zmenách tendenciu degradovať. Veľké zmeny softvéru môžu byť časovo a finančne náročné. Zákazníci môžu komentovať a vidieť, koľko práce už bolo vykonanej.
- **Orientované na opätovné použitie (*Reuse-oriented*)** – Generický vzor, ktorý sa v súčasnosti stal dominantnou paradigmou pri vývoji webových informačných a podnikových systémoch. Ide o opätovné využitie už naprogramovaných komponent, prípadne celých balíčkov softvéru. Príkladom je softvérový *framework* – univerzálny a znovu-použiteľný softvér, ktorý sa vyžíva pri vývoji aplikácií, produktov a rôznych riešení. Softvérový *framework* by mal obsahovať podporné knižnice, kompilátory, komponenty, ale hlavne API (*Application programming interface* – rozhranie pre programovanie aplikácií). [8]

V praxi sú najväčšie systémy vytvorené použitím procesu, ktorý zahŕňa všetky zo spomínaných troch generických vzorov.

## 5.2 Analýza

Softvérová analýza v sebe zahŕňa aktivity, pri ktorých sa určia softvérové procesy, objekty a vzťahy medzi nimi.

Pred samotným návrhom je dôležité určiť model softvérového procesu, ktorý bude počas vývoja použitý, podobne aj definovať funkčné, respektíve nefunkčné požiadavky na systém. Na vývoj systému **Obedár** bol použitý inkrementálny vývoj, pričom som využíval už existujúce komponenty a teda aj *reuse-oriented* prístup.

### 5.2.1 Stakeholders

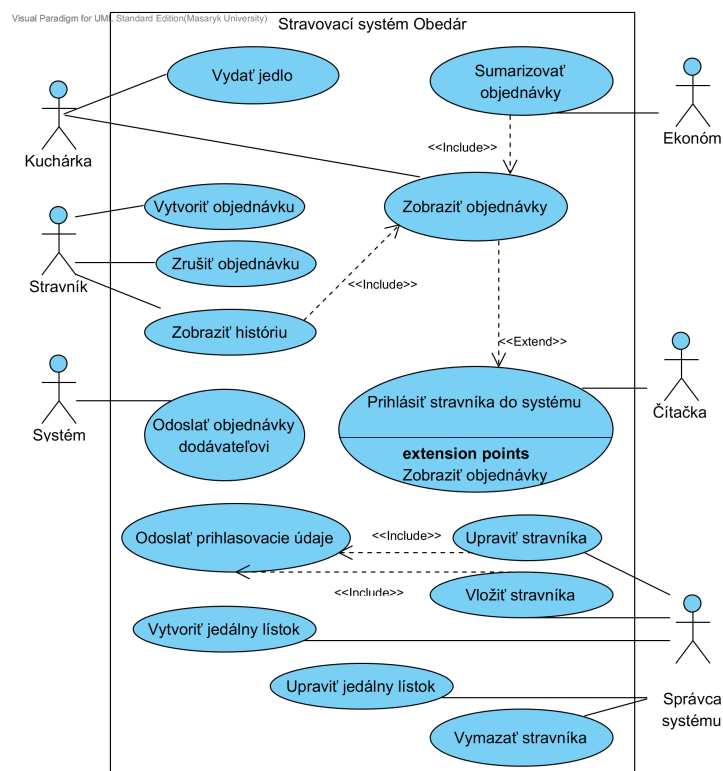
Počas analýzy som sa stretol so zainteresovanými osobami, ktorí mali určitý vplyv na vývoj stravovacieho systému alebo boli vývojom ovplyvnení. Od nich záviselo celkové porozumenie požiadavkám na systém. Táto skupina ľudí sa zvykne označovať ako *stakeholders* – zainteresované osoby:

- **riaditeľ** ako schvaľovateľ nového stravovacieho systému
- **stravníci**, ktorých objednávky sú zaznamenané v systéme
- **manažéri** dohliadajúci na správny chod spoločnosti. V systéme fungujú ako obyčajní stravníci a nemajú špeciálne privilégia.
- **správca IT** zodpovedný za správne fungovanie hardvéru a webového serveru
- **kuchárka** ovládajúca systém pri výdaji jedál
- **ekonómka**, ktorá má na starosti finančné prostriedky

Funkčné požiadavky

Špecifikácia funkčných požiadaviek bola po vytvorení *lean canvas* modelu (pozri 4.1) už jednoduchá.

**Use case Diagram** – diagram prípadov, slúži na reprezentovanie funkčných požiadavkov na informačný systém. Tvorí ho hranica systému, aktéri, prípady použitia a vzťahy. Aktéri predstavujú role, ktoré nadobúda externá entita. V princípe sa môže jednať o systém, fyzickú osobu alebo aj čas. Jedna fyzická osoba môže nadobúdať viacero rolí.



Obr. 5.1: UseCase Diagram

Pre **stravníkov** existujú dva spôsoby vytvorenia objednávky v systéme. Objednávkou sa rozumie každé záväzné objednanie jedla na určitý deň. Každému **stravníkovi** je umožnené túto objednávku zrušiť najneskôr do začiatku dňa, pre ktorý bola objednávka vytvorená. Stravník môže vytvoriť objednávku využitím počítača s dotykovou obrazovkou v kantíne alebo cez webové rozhranie. Prihlásenie v kantíne prebieha pomocou identifikačnej kartičky. V prípade webového rozhrania sa stravník musí autentizovať svojim používateľským menom a heslom. Stravník môže byť vytvorený ekonómkou, pričom sa mu vygeneruje používateľské meno z priezviska a heslo z náhodnej postupnosti znakov.

**Kuchárka** má kompetencie používať stravovací systém s vyššími privilégiami ako ostatní zamestnanci. Má za úlohu vydávať stravníkom príslušné jedlá podľa vytvorených objednávok. Je taktiež schopná zobrazíť zostávajúce – nespracované objednávky pre každého stravníka.

**Ekonómka** má možnosť zobrazíť sumarizáciu objednávok s daným roz-



sahom dní. Zobrazí sa tak štatistika objednávok. Ekonómka má v systéme prístup k sumarizácii objednaných jedál každého stravníka a podľa toho odpočítava peniaze z finančnej odmeny stravníkom. Má právo vytvoriť nového stravníka v systéme. Pri vytvorení zadáva povinné údaje ako meno, priezvisko, email. Taktiež registruje RFID kartičku pre stravníka. Má možnosť vymazať a opätovne obnoviť stravníka. Vymazaný stravník sa nemôže prihlásiť do systému.

**Manažéri** sa starajú o komunikáciu s externou firmou, ktorá dodáva hotové jedlá.

**Čítačka** má za úlohu prihlásiť stravníka do systému

**Systém** predstavuje aktéra, pretože je potrebné odosielať objednávky dodávateľovi.

Nefunkčné požiadavky

**Autentizácia** – Systém podporuje RFID (*Radio-frequency identification*) s nosnou frekvenciou **125 kHz**. Slúžia k bezkontaktnnej komunikácii. Stravníci majú možnosť objednávať jedlá aj diaľkovo cez internet. Preto musia existovať dva spôsoby priebehu autentizácie.

Prvý spôsob je autentizácia na diaľku pomocou prideleného používateľského mena a hesla. Používateľ nemusí byť na lokálnej sieti ale prihlasovať sa využitím *HTTP* protokolu.

Druhý spôsob je taký, že sa stravník prihlási do systému pomocou svojej jedinečnej kartičky podporujúcej štandard RFID.

**SMTP** – Systém je schopný komunikovať so smtp serverom a odosielať emaily stravníkom, ktoré sú potrebné pri prvotnej registrácii. Rovnako musí systém odosielať email dodávateľskej firme, ktorá jedlá pripravuje.

## 6 Návrh IS

Analýza a návrh systému sú aktivity, ktoré sa dosť prekrývajú a nie je možné s istotou identifikovať, či sa ešte vykonáva analýza alebo už ide o návrh systému.

V kapitole sa venujem návrhu systému pomocou diagramu tried a návrhu entít databáze. Fáza, v ktorej sa rozhoduje, akým spôsobom by mali byť funkcie systému implementované. Opísať však úplne všetky triedy a spôsoby implementácie je nad optimálny rozsah tejto práce.

Návrh informačného systému predchádza pred samotnou implementáciou. Vďaka návrhu je možné utvoriť si detailný obraz o systéme z hľadiska implementácie.

### 6.1 Diagram tried

Špecifikácia návrhových tried by mala byť na takej úrovni, aby vývojár bol schopný naprogramovať systém bez väčších konzultácií s architektom systému. Je teda nutné presne špecifikovať správanie triedy v systéme.

Na zachytenie tried v navrhovanom systéme som použil diagram tried – *class diagram*. Spadá do množiny UML (*Unified Modeling Language*). Zobrazuje triedy objektov v systéme a vzťahy medzi nimi. [11, s. 129].

Triedy sú prepojené **asociáciami** – prepojenie medzi triedami indikujúce nejaký vzťah, pričom sa používa multiplicita (číslo pri asociácii, ktoré hovorí o kardinalite množiny).

**Generalizácia** sa v diagrame značí šípkou, ktorá smeruje od špecifickej triedy smerom ku všeobecnej triede.

Avšak, na dôkladnú špecifikáciu celého systému nie je dostatočný priesor, preto som zachytil len niektoré „kľúčové“ triedy, pomocou ktorých si je možné urobiť obraz o systéme.

V diagrame tried (obrázok 6.1) sú objekty, ktoré slúžia na akúsi vrstvu medzi biznis logikou systému a dátovou vrstvou. Biznis logiku reprezentujú manažérske triedy, ktoré majú v názve sufix „*Manager*“. Tieto manažérske triedy slúžia na operácie s objektmi, rôzne výpočty atď. Dátovú vrstvu prezentujú objekty ako *Meal*, *Order* atď. Od týchto objektov je následne odvodená databáza celého systému.



## 6.2 Databáza

Ideálny stav by sa dal nazvať vtedy, pokiaľ pri ďalšom vývoji zostane štruktúra databázy rovnaká. Docieliť niečo také je v praxi takmer nemožné. Preto pri inkrementálnom vývoji (pozri 5.1), má navrhnutý softvér tendenciu degradovať.

### 6.2.1 ERD (Entity–Relationship diagram)

Databáza môže byť modelovaná ako kolekcia entít, ktoré sú spolu pospájané rôznymi vzťahmi.

#### Entita

Existujúci objekt, ktorý je odlišiteľný od iných objektov. Môže sa jednať o špecifickú osobu, organizáciu, udalosť atď. Entita obsahuje nejaké atribúty.

Kľúče slúžia na jednoznačnú identifikáciu entity. V mojom databázovom modeli má každá entita kľúč so sufixom „Id“.

#### Entitná množina

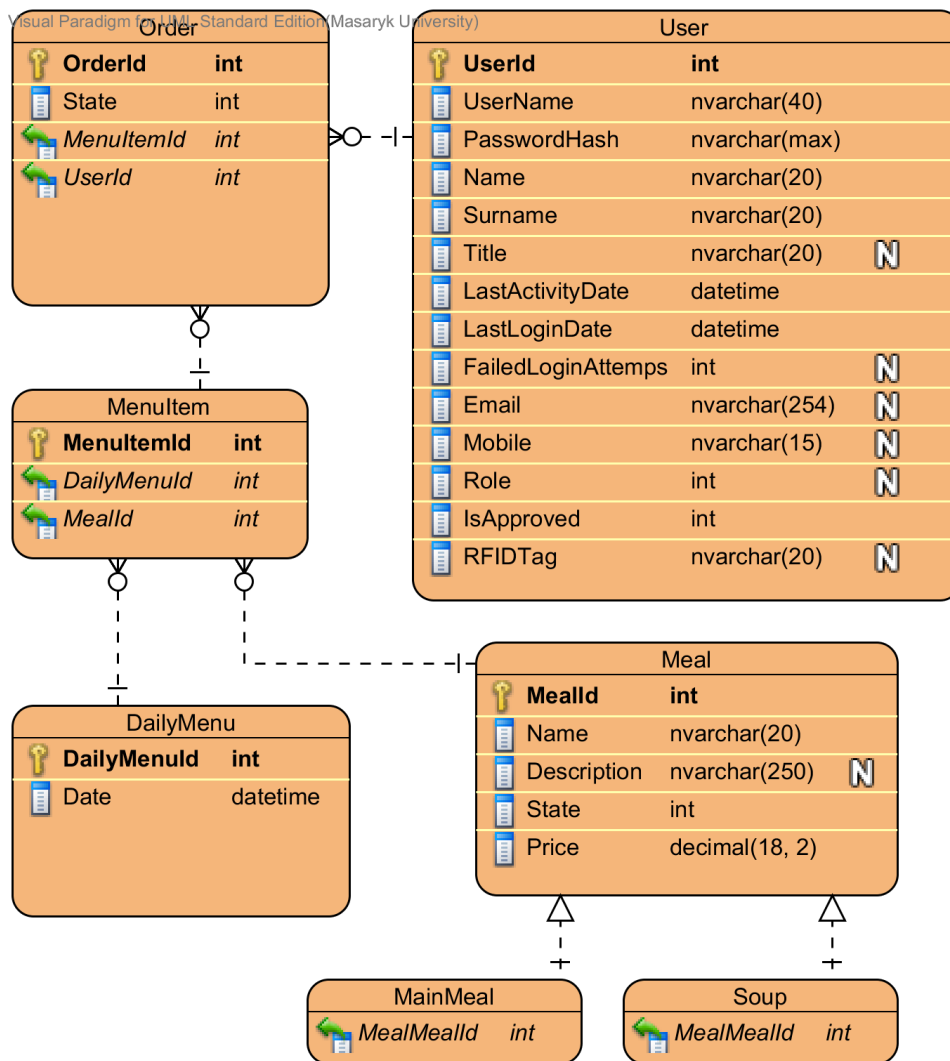
Množina, ktorá obsahuje entity rovnakého typu a zdieľa rovnaké vlastnosti. Množina všetkých používateľov, organizácii a iné. [9, s. 3]

#### Relácia

Spojenie medzi niekoľkými entitami. Používajú sa kardinálne obmedzenia podobne ako pri diagrame tried.

#### Početnosť relácii

Označuje počet entít, s ktorými môžu byť ostatné entity prepojené pomocou množiny vzťahov. Najužitočnejšie sú v popise binárnych relácii. Pre binárne množiny vzťahov musí byť početnosť v pomere 1:1, 1:N, N:1 alebo N:M. [9, s. 11]



Obr. 6.2: Entity–Relationship diagram

## 7 Implementácia

Vďaka analýze a návrhu informačného systému, ktoré som pripravil v predchádzajúcich kapitolách môžem prejsť na ďalší krok. Týmto krokom je implementácia. Fáza v sebe zahŕňa voľbu vhodných technológií spomedzi veľkého množstva, ktoré sú v súčasnosti k dispozícii.

Postupne opíšem softvér a technológie, ktoré som použil na vývoj informačného systému **Obedár**.

Na vývoj systému používam vývojové prostredie *Visual Studio 2012* od spoločnosti *Microsoft*. Systém som rozdelil na 3 projekty.

- **Obedar** – kompletný systém vytvorený ako webová aplikácia
- **Obedar.Scheduler** – konzolová aplikácia, ktorá slúži na pravidelné zasielanie objednávok dodávateľskej firme.
- **Obedar.Tests** – obsahuje testy systému (pozri 8)

### 7.1 Použité technológie

Z predchádzajúcich kapitol je zrejmé, že systém bude vyvinutý ako webová aplikácia. Technológie, ktoré opisujem v tejto sekcii sú vybudované na platforme **.NET Framework**. Využíva sa na vývoj softvéru pre operačné systémy Windows od spoločnosti Microsoft.

#### 7.1.1 ASP.NET

Webový aplikačný *framework* navrhnutý pre vývoj webových aplikácií. Umožňuje vytvárať dynamické webstránky, webové aplikácie ale aj webové služby.

Existujú 3 programovacie modely, ktoré sa používajú pri vývoji v technológii ASP.NET:

1. Web Pages
2. Web Forms
3. MVC

Keď bol prvýkrát vydaný ASP.NET verzie 1,0 v roku 2002, bolo veľmi ľahké zamieňať ASP.NET a *Web Forms* za tie isté technológie. Avšak už od samotného počiatku návrhu bola platforma ASP.NET navrhnutá s vrstvami

abstrakcie a to vďaka nasledujúcemu rozdeleniu menných priestorov: [1, s. 2]

- **System.Web.UI** – vrstva pre **Web Forms** model, serverové komponenty a iné.
- **System.Web** – obsahuje triedy na podporu komunikácie medzi klientom (internetový prehliadač) a serverom. Obsahuje moduly, *handlers*, *HTTP* zásobník a iné.

## MVC

Celým názvom **ASP.NET MVC** implementuje známu softvérovú architektúru *Model-View-Controller* (MVC), vďaka ktorej je možné efektívne oddeliť dátovú časť, užívateľské rozhranie a riadiacu logiku. Bol vytvorený na rozdelenie komponentov používateľského rozhrania.

## Web Forms

Princíp spočíva vo webstránkach, ktoré majú vytvorený životný cyklus. V tomto cykle sa volajú rôzne udalosti od inicializácie, cez načítanie stránky a viazanie dát až po etapu vykresľovania. Stránky sa kompilujú a spúšťajú na serveri. Následne vygenerujú výstupné *HTML*, ktoré sa zobrazí klientovi. Po dôkladnej analýze som zvolil práve túto technológiu.

### 7.1.2 Entity Framework

**.NET Framework** má implementovanú technológiu **ADO.NET**, ktorá umožňuje prístup k dátam a dátovým službám (napr. *SQL Server*). Obsahuje triedy, pomocou ktorých je možné vytvoriť spojenie na rôzne dátové zdroje a následne vykonávať príkazy a operácie na dátach. **Entity Framework** (EF) podporuje vývoj dátovo-orientovaných softvérových aplikácií. Vďaka EF je možné databázové objekty (tabuľky) preniesť do objektov v .NET a pracovať s nimi. Databáza sa pomocou EF vytvára dvoma rôznymi prístupmi:

- *model first* – umožňuje navrhovať databázu v dizajnerskom móde. Vývojár navrhne databázu podobným spôsobom ako pri modelovaní ERD (pozri 6.2.1).
- *code first* – návrh databázy spočíva vo vytvorení tried, ktoré obsahujú vlastnosti. Vďaka anotáciám dokáže EF zistiť, či má pre danú triedu vytvoriť tabuľku v databáze. Ja som zvolil práve tento prístup.

Podľa ERD diagramu, ktorý som navrhol v časti 6.2.1 má databáza obsahovať tabuľku s názvom **User**. Kód, z ktorého EF vygeneruje tabuľku som načrtol v ukážke 7.1:

Ukážka 7.1: EF – tabuľka User

```
[System.ComponentModel.DataAnnotations.Key]
[System.ComponentModel.DataAnnotations.Schema.Table("User")]
namespace Obedar.Model
{
    [System.ComponentModel.DataAnnotations.Schema.Table("User")]
    public class User
    {
        [DatabaseGeneratedAttribute(DatabaseGeneratedOption.Identity)]
        public int UserId { get; set; }

        ...

        public virtual System.Collections.Generic.ICollection<Obedar.Model.Order> Orders { get; set; }
    }
}
```

Kľúčové slovo *virtual* pred kolekciou s názvom *Orders* zabezpečí vytvorenie vzťahu medzi tabuľkou *Order* a *User*. Tabuľka *Order* bude obsahovať cudzí kľúč na tabuľku *User*.

### Kontext databáze

*DbContext* je trieda, ktorá implementuje interakciu dát ako s objektmi .NET. Obsahuje kolekcie „tabuliek“, sleduje zmeny, ktoré boli na objektoch vykonané, spravuje súbežnosť a iné. [4]

Vytvoril som vlastnú triedu, ktorá dedí zo spomínanej *DbContext*. Ďalej stačilo už len vytvoriť vlastnosti triedy – kolekcie objektov, ktoré sa mapujú na tabuľky v databáze.

Ukážka 7.2: EF – kontext databáze

```
public class ObedarContext : System.Data.Entity.DbContext
{
    ...
    public System.Data.Entity.DbSet<Obedar.Model.User> Users { get; set; }
    ...
}
```

V prípade zavolania metódy *get* sa automaticky vygeneruje SQL dotaz na databázu.



### 7.1.3 LINQ

*Language-Integrated Query* (LINQ) je súčasť platformy .NET verzie 3.5 a viac. Zjednodušuje a zjednocuje implementáciu ľubovoľného prístupu k dátam, pričom nie je nutné používať presne špecifikovanú architektúru. Preto existujú rôzne nadstavby na LINQ: [3]

- LINQ to Objects
- LINQ to XML
- LINQ to ADO .NET
  - LINQ to DataSet
  - LINQ to SQL
  - LINQ to Entities

#### LINQ to Entities

Technológiu *LINQ to Entities* som pri vývoji používal najčastejšie.

Poskytuje podporu LINQ pre EF. Umožňuje pracovať s dátami, prípadne štruktúrou databáze použitím štandardných .NET objektov. Princíp je taký, že sa vytvorí LINQ dotaz napríklad na kolekciu Users definovanej v triede, ktorá dedí z *DbContext*. LINQ dotaz je následne preložený do klasického SQL dotazu a spustený v momente, keď sú dáta potrebné.

Ukážka 7.3 zobrazuje syntax dotazovania na objekty, ktoré sú uložené v databáze. *Orders* je kolekcia definovaná v triede *ObedarContext*. Výstup bude kolekcia objektov typu *StatisticsModel* – podporná trieda, ktorá obsahuje vlastnosti: identifikátor používateľa, počet objednávok a finálnu cenu.

Ukážka 7.3: Linq to Entities – Query syntax

```

1  var results = from o in Orders
2      where o.MenuItem.DailyMenu.Date > new DateTime("2013-04-1  ↵
3          00:00:00") &&
4          o.MenuItem.DailyMenu.Date < new DateTime("↵
5              2013-04-30 00:00:00")
6      group o~by o.User.UserId into g
7      select new Obedar.Model.StatisticsModel()
8      {
9          UserId = g.Key,
10         FinalPrice = g.Sum(o => o.MenuItem.Meal.Price),
11         NumberOfOrders = g.Count()
12     };

```

V ukážke 7.3 na 9. riadku je použitý lambda výraz. Lambda operátor v jazyku C# sú dva znaky: `=>`. V kóde využívam už naimplementovanú metódu `Sum()`, ktorá mi vďaka lambda výrazu umožní spočítať cenu objednávok. Vďaka *LINQ to Entities* teda môžem vytvoriť aj zložitejší dotaz na jeden riadok oproti písaniu dlhého skriptu SQL, prípadne procedúry.

#### 7.1.4 Responsive web design

*Responsive web design* (RWD) – spôsob tvorby dizajnu webstránky, ktorý zaručí, že dizajn bude optimalizovaný pre rôzne druhy zariadení. Vďaka novej verzii *Cascading Style Sheets* (CSS3) je možné rozpoznať vlastnosti zariadení, na ktorom je stránka zobrazená a prispôbiť tak samotnú stránku a jej obsah. [7]

Vývoj aplikácii pre mobilné platformy by sa z hľadiska slabej využiteľnosti dalo chápať ako plytvanie zdrojmi (pozri 4.2.1). V systéme **Obedár** som využil aspoň RWD na prispôbenie dizajnu pre zariadenia s nízkym rozlíšením.

## 7.2 Informačný systém Obedár

Celý systém je implementovaný ako webová aplikácia v podprojekte s názvom **Obedar**. Zvolená technológia je **ASP.NET WebForms** pri využití EF a *Linq to SQL* technológií.

### 7.2.1 Štruktúra

Každý informačný systém by mal mať zdrojové súbory uložené v priečinkoch závislosti na svojej vzájomnej podobnosti. Ja som použil už existujúce priečinky, ktoré boli vytvorené pomocou vývojového prostredia. Dôvodom je, že niektoré priečinky ako *AppStart*, *Content*, *Model* a iné, tvoria súčasťou platformy ASP.NET [1, 26]. Tabuľka 7.1 obsahuje zoznam priečinkov a stručný popis, aké súbory sú v nich umiestnené.

#### Web.config

Hlavný konfiguračný súbor ASP.NET webovej aplikácii. Rovnako ako webstránky je uložený v koreňovom priečinku projektu. Obsahuje rôzne nastavenia bezpečnosti, nastavenia databázového servera, vlastné nastavenia aplikácie a iné. *Web.Config* sa dá upravovať už na produkčnom servery. Nie je potrebné kvôli zmene celý systém nasadiť na server.

| NÁZOV             | POPIS                                                                                                  |
|-------------------|--------------------------------------------------------------------------------------------------------|
| <b>AppStart</b>   | Miesto pre konfiguračný kód.                                                                           |
| <b>Content</b>    | CSS súbory a iný stránkový obsah.                                                                      |
| <b>Controls</b>   | Komponenty, ktoré sa využívajú vo webstránkach. De-<br>dia z triedy <i>System.Web.UI.UserControl</i> . |
| <b>Email</b>      | Triedy umožňujúce komunikáciu emailom.                                                                 |
| <b>Images</b>     | Obrázky použité na webstránke.                                                                         |
| <b>Managers</b>   | Manažérske triedy predstavujúce biznis logiku systému.                                                 |
| <b>Membership</b> | Triedy, ktoré spravujú členstvo, prihlásenie do sys-<br>tému, role používateľov atď.                   |
| <b>Migrations</b> | Obsahuje len triedu <i>Configuration</i> , ktorá pripraví pilotnú<br>databázu počas <i>debug</i> .     |
| <b>Model</b>      | Miesto na triedy, ktoré reprezentujú dáta a manipulujú<br>s nimi.                                      |
| <b>Parameters</b> | Parametre pre jednotlivé webstránky                                                                    |
| <b>Scripts</b>    | Knižnice a skripty <i>JavaScript</i> .                                                                 |
| <b>Security</b>   | Metódy slúžiace na vygenerovanie hesla.                                                                |

Tabuľka 7.1: Priečinky v systéme [1, 26]

| NÁZOV              | POPIS                                                                                               |
|--------------------|-----------------------------------------------------------------------------------------------------|
| <b>History</b>     | Zobrazuje históriu objednávok pre stravníka.                                                        |
| <b>Login</b>       | Slúži na vzdialené prihlásenie do systému. (pozri 7.2.2)                                            |
| <b>Users</b>       | Zobrazuje zoznam všetkých stravníkov v systéme, pri-<br>čom sú k dispozícii rôzne akcie k editácii. |
| <b>UserDetails</b> | Slúži na vytvorenie alebo editovanie stravníka.                                                     |
| <b>Meals</b>       | Zoznam všetkých jedál.                                                                              |
| <b>MealDetails</b> | Slúži na vytvorenie alebo editovanie jedla v systéme                                                |
| <b>Menus</b>       | Umožňuje vytvárať a editovať jedálne lístky.                                                        |
| <b>Orders</b>      | Slúžia na vytváranie objednávok pre stravníkov.                                                     |
| <b>Statistics</b>  | Zobrazuje sumarizáciu zakúpených jedál pre každého<br>stravníka za zvolené obdobie.                 |
| <b>Chef</b>        | Zobrazuje aktuálne objednávky stravníka po prihlásení<br>pomocou RFID kartičky                      |
| <b>LocalOrders</b> | Umožňuje vytvárať a editovať objednávky pre stravní-<br>kov na lokálnej sieti.                      |

Tabuľka 7.2: Webstránky v systéme

## Webstránky

Samotné webstránky sú uložené priamo v koreňovom priečinku projektu **Obedar**. Vďaka nim je možné manipulovať s dátami v databáze. V tabuľke 7.2 stručne opisujem ich funkcie. Príloha C obsahuje ukážky niektorých webstránok systému.

### 7.2.2 Bezpečnosť

V tejto sekcii sa venujem popisu ako je implementovaná bezpečnosť v systéme. Rovnako opíšem role systému a spôsob prihlásenia do systému.

Webstránky, ktoré umožňujú prihlásenie pomocou RFID kartičky nemôžu byť prístupné z vonkajšej siete. Stránky, ktoré umožňujú prihlásenie cez RFID sú *Chef.aspx* a *LocalOrders.aspx*. Prístupné sú preto len z lokálnej siete a zo špecifickej IP adresy. Túto úlohu má na starosti správca IT v spoločnosti. Bezpečnostné nastavenia je možné konfigurovať v spomínanom súbore *Web.Config* (pozri 7.2.1).

#### Používateľské role

Používatelia môžu v systéme vystupovať v roli zamestnanca alebo administrátora.

**Zamestnanec** – rola v systéme, ktorú má každý stravník. Má vzdialený prístup k týmto stránkam: *Login.aspx*, *Orders.aspx*, *History.aspx*.

**Administrátor** – Podľa predvoleného nasadenia systému je administrátor len jeden. Z analýzy funkčných požiadaviek vyplýva, že tieto privilégia má ekonómka (pozri 5.2.1). Ekonómka sa prihlasuje do systému ako administrátor. Administrátor má prístup takmer ku všetkým stránkam systému, výnimkou sú *Orders.aspx* a *History.aspx*.

#### Autorizácia

Systém nie je prístupný verejnosti, a teda nie je možné získať akýkoľvek prístup k dátam bez overenia identity. Pre zobrazenie akejkoľvek webstránky musí byť používateľ autorizovaný.

V ukážke 7.4 zobrazujem jednoduché nastavenie v konfiguračnom súbore *Web.Config*, ktoré zakáže fyzický prístup k stránke *UserDetails.aspx* všetkým používateľom okrem tých, ktorí v systéme vystupujú v roli Administrátor.

## Ukážka 7.4: Obmedzenie prístupu pre UserDetails.aspx

```
<location path="UserDetails.aspx">
  <system.web>
    <authorization>
      <allow roles="Administrator" />
      <deny users="*" />
    </authorization>
  </system.web>
</location>
```

## Prihlásenie

Trieda *Membership* poskytuje množinu statických metód a statických vlastností pre prístup k používateľom, prípadne k rolám používateľov. Tieto metódy pracujú s ďalšou triedou s názvom *MembershipProvider*. Všetky triedy, ktoré majú na starosti členstvo, sa nachádzajú v mennom priestore *System.Web.Security*.

Systém musel byť implementovaný tak, aby bolo stravníkom umožnené prihlásiť sa dvoma rôznymi spôsobmi. Kvôli tejto požiadavke som vytvoril vlastný objekt, ktorý nahrádza klasický *membership provider*. *Membership API* je založená na tzv. *Forms* autentizácii a poskytuje infraštruktúru pre manažment a autentizáciu používateľov.

Môj vlastný provider dokáže prihlásiť používateľa prečítaním RFID karty čítačkou (prístup je jedine z lokálneho počítača s presne definovanou sieťovou adresou) alebo zadaním prihlasovacieho mena a hesla (využitím hlavnej stránky *Login.aspx*). Tento provider je implementovaný ako trieda s názvom *ObedarMembershipProvider*. Trieda dedí z abstraktnej triedy *MembershipProvider*. Bolo potrebné preťažiť niektoré metódy, ktoré klasický *membership provider* poskytuje.

### 7.3 Obedar.Scheduler

Projekt *Obedar.Scheduler* je konzolová aplikácia vyvinutá taktiež pod .NET platformou. Úloha tejto aplikácie je odosielať každý sumarizáciu objednávok dodávateľskej spoločnosti. Aplikácia odosiela email s potrebnými informáciami pomocou SMTP protokolu.

Po použití rôznych technológií a implementácii celého systému som pokračoval testovaním. V ďalšej kapitole opisujem spôsob testovania, ktorý bol použitý pri tvorbe systému.

## 8 Testovanie

Každý informačný systém pred produkčným spustením by mal byť dostatočne otestovaný. Testovanie je súčasť vývojového cyklu informačného systému.

### 8.1 Unit Test

*Unit Test* je metóda testovania, pri ktorej sa overí správna funkčnosť individálnych častí systému. Tieto časti v objektovo-orientovaných jazykoch predstavujú triedy. Princíp vývoja *Unit* testov spočíva v tom, že vývojár sa pokúsi zavolať nejakú metódu testovanej triedy a overí výsledok tejto metódy s pôvodným očakávaným výsledkom. Pokiaľ výsledok nie je zhodný s predpokladmi, tak test zlyhá.

Testy majú overovať správnu funkčnosť čo najmenších častí systému. *Unit Test* musí byť rýchly a automatický. Unit testy sú najčastejšie umiestnené v oddelenom projekte.

#### 8.1.1 NUnit framework

Patrí medzi najpoužívanejší testovací *framework*. NUnit je testovací framework pre všetky programovacie jazyky spadajúce do platformy .NET. Bol vyvinutý v jazyku C# [6]. Ďalší príklad využitia už existujúcich komponentov – opäť som použil *reuse-oriented* generický vzor (pozri 5.1).

Pri pomenovaní testov je žiadúce dodržiavať nasledujúce konvencie:

- projekt – meno testovaného projektu spojený so sufixom „Tests“
- trieda – meno testovanej triedy so sufixom „Tests“
- metóda – meno testovanej metódy + stav pri testovaní + očakávaný výsledok

Vytvoril som projekt **Obedar.Tests**, ktorý slúži na testovanie biznis logiky systému. V ňom sa nachádzajú testovacie triedy, pri ktorých využívam NUnit framework.

## 9 Záver

Cieľom tejto práce bolo navrhnuť a implementovať informačný systém pre stravovacie zariadenie.

Pôvodne som predpokladal, že tvorba informačného systému nemôže byť príliš náročná úloha. Veľmi rýchlo som však pochopil, že každá spoločnosť má svoje vlastné požiadavky na systém a veľakrát tieto požiadavky ani nedokážu presne definovať. Kvôli tejto skutočnosti je potrebné pred samotným vývojom zanalyzovať skutočné problémy, ktoré vznikajú a prideliť im adekvátnu prioritu. Práve preto považujem využitie metodiky *lean* za správny krok. Vykonal som prieskum medzi zamestnancami, vďaka čomu som si mohol urobiť presnejšiu predstavu o systéme.

Postupne som vykonal analýzu, návrh a implementáciu nového IS. Následne na to bolo nutné otestovať správnu funkcionality IS.

Výsledkom tejto práce je informačný systém s názvom **Obedár**, ktorý zlepšuje a zrýchľuje procesy v stravovacích zariadeniach. Rovnako dôležitá je aj skutočnosť, že systém sa reálne využíva.

Po konzultácii so zamestnancom, pánom Bc. Lukášom Cabajom, ktorý má v spoločnosti **DOR s.r.o.** na starosti IT zariadenia, bolo zhodnotené, že systém **Obedár** spĺňa požiadavky a je vhodný na spustenie do plnej prevádzky. Avšak, kvôli nedostatku času bol systém zatiaľ spustený v testovacom režime.

Do budúca bezpochyby plánujem vyvíjať ďalšie rozšírenia a nové funkcie v závislosti na zmene požiadaviek, prípadne po príchode nových zákazníkov. Veľký potenciál vidím vo využití *cloud* služieb, vďaka čomu by spoločnosti nemuseli kupovať vlastné servery, licencie atď.

Pri väčšom počte zamestnancov, ktorí disponujú s modernými mobilnými telefónmi, by som rád vývoj zamerlal na využitie technológie *Near field communication* (NFC). Čo by umožnilo autentizáciu pomocou mobilných telefónov.

Vďaka tejto práci som si mimoriadne zlepšil svoje vedomosti o platforme .NET. Naučil som sa množstvo nových vecí v oblasti programovania ale aj v organizovaní. Porozumel som, aká dôležitá je komunikácia so zainteresovanými ľuďmi, vďaka ktorej vývojár získava spätnú väzbu. Aj preto je pre mňa táto práca veľkým prínosom.

## Literatúra

- [1] GALLOWAY, J.; HAACK, P.; WILSON, B.; aj.: *Professional ASP.NET MVC 4*. Wiley, prvé vydanie, 2012, ISBN 1-118-34846-X, ix, 440 s.
- [2] MAURYA, A.: *Running Lean*. Sebastopol: O'Reilly Media, druhé vydanie, 2012, ISBN 1-449-30517-2, 240 s.
- [3] Microsoft: *LINQ and ADO.NET*. Microsoft Developer Network. Microsoft, c2013, [Online, cit. 2.3.2013]. Dostupné na World Wide Web: <http://msdn.microsoft.com/en-us/library/vstudio/bb399365.aspx>
- [4] Microsoft: *Working with DbContext*. Microsoft Developer Network. Microsoft, c2013, [Online, cit. 2.3.2013]. Dostupné na World Wide Web: <http://msdn.microsoft.com/en-us/data/jj729737>
- [5] MLEJNKOVÁ, L.: *Služby společného stravování*. Praha: Oeconomica, druhé vydanie, 2009, ISBN 8-024-51592-X, 130 s.
- [6] POOLE, C.: *What Is NUnit?*. Charlie Poole, c2012, [Online, cit. 2.5.2013]. Dostupné na World Wide Web: <http://nunit.org/index.php?p=home>
- [7] Prispievatelia: *Responzivní web design*. Wikipedie: Otvorená encyklopedie. c2013, [Online, cit. 4.05.2013]. Dostupné na World Wide Web: [http://cs.wikipedia.org/w/index.php?title=Responzivn%C3%AD\\_web\\_design&oldid=10143874](http://cs.wikipedia.org/w/index.php?title=Responzivn%C3%AD_web_design&oldid=10143874)
- [8] Prispievatelia: *Software framework*. Wikipedia, The Free Encyclopedia. c2013, [Online, cit. 20.4.2013]. Dostupné na World Wide Web: [http://en.wikipedia.org/w/index.php?title=Software\\_framework&oldid=550544433](http://en.wikipedia.org/w/index.php?title=Software_framework&oldid=550544433)
- [9] SILBERSCHATZ, A.; KORTH, H.; SUDARSHAN, S.: *Database Design: The Entity-Relationship Approach*. Silberschatz, Korth and Sudarshan, c2010, [Online, cit. 29.4.2013]. Dostupné na World Wide Web: <http://codex.cs.yale.edu/avi/db-book/db6/slide-dir/PDF-dir/ch7.pdf>
- [10] SOMMERVILLE, I.: *Software Engineering*. Harlow: Addison-Wesley, 6 vydanie, 2001, ISBN 0-201-39815-X, xx, 693 s.



- [11] SOMMERVILLE, I.: *Software Engineering*. Harlow: Addison-Wesley, 9 vydanie, 2010, ISBN 0-13-703515-2, 792 s.
- [12] ŠMÍD, V.: *Pojem informačního systému*. c2013, [Online, cit. 12.4.2013]. Dostupné na World Wide Web: <http://www.fi.muni.cz/~smid/mis-infsys.htm>

## A Obsah CD

K práci je priložené CD, ktoré obsahuje nasledujúce údaje:

- zdrojové súbory informačného systému **Obedár**
- inštalačné pokyny ako systém sprevádzkovať
- textovú časť práce vo formáte *Portable Document Format* (PDF)

## B Lean Canvas Model

| PROBLEM                                                                                                                                   | SOLUTION                                                                                                                                                  | UNIQUE VALUE PROPOSITION                                                                                                   | UNFAIR ADVANTAGE | CUSTOMER SEGMENTS                                                                                                            |
|-------------------------------------------------------------------------------------------------------------------------------------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------|----------------------------------------------------------------------------------------------------------------------------|------------------|------------------------------------------------------------------------------------------------------------------------------|
| Rad v jedálni.<br><br>Fyzická prítomnosť na vytváranie objednávok.<br><br>Kontrola objednávok<br><br>EXISTING ALTERNATIVES<br><br>ObedDOR | História objednávok<br><br>Webové rozhranie                                                                                                               | Jednoduché a rýchle objednávanie jedál pre zamestnancov z pohodlia domova.<br><br>HIGH-LEVEL CONCEPT<br><br>Webový ObedDOR | História dát     | Firmy<br><br>Školy<br><br>Stravníci v zamestnaneckej stravovacej prevádzke - robotníci, ekonómovia, sekretárky, riaditeľ ... |
| COST STRUCTURE<br><br>Mesiac intenzívneho vývoja produktu ~ 1500€<br><br>Meetingy so stakeholders ~ 100€<br><br>Softvér ~ 615€            | KEY METRICS                                                                                                                                               |                                                                                                                            | CHANNELS         | EARLY ADOPTERS<br><br>Firmy s jedáňou a zamestnaneckým stravovaním.                                                          |
|                                                                                                                                           | REVENUE STREAMS<br><br>Prvotná inštalácia a zavedenie systému 199€, potom 99€/rok<br><br>Zaškolenie zamestnancov – 9,99€/osoba<br><br>Servis – 24,99€/hod |                                                                                                                            |                  |                                                                                                                              |

Lean Canvas is adapted from The Business Model Canvas (BusinessModelGeneration.com) and is licensed under the Creative Commons Attribution-Share Alike 3.0 Unported License.

## C Ukážky používateľského rozhrania

### Login.aspx – Prihlásenie

 **OBEDÁR**

Prihlásiť

Domov

 **Prihlásenie**

Používateľské meno:

Heslo:

☒ Pamätať si?

Prihlásiť

### Users.aspx – Zamestnanci

 **OBEDÁR**

Vítajte, administrator !

Odhlásiť

Domov Zamestnanci Jedlá Jedálne lístky Štatistiky

**Zamestnanci**

 Pridať zamestnanca

| Id | Login         | Meno         | Priezvisko   | Email                  | Mobil         | Posledná aktivita  | Akcie |
|----|---------------|--------------|--------------|------------------------|---------------|--------------------|-------|
| 1  | administrator | Jakub        | Adamus       | jakob.adamus@gmail.com | +421949478605 | 14.5.2013 22:57:53 |       |
| 2  | zamestnanec0  | zamestnanec0 | zamestnanec0 | zamestnanec0@mail.sk   |               | 14.5.2013 20:30:30 | ✕ 🔑   |
| 3  | zamestnanec1  | zamestnanec1 | zamestnanec1 | zamestnanec1@mail.sk   |               | 1.1.0001 0:00:00   | ✕ 🔑   |
| 4  | zamestnanec2  | zamestnanec2 | zamestnanec2 | zamestnanec2@mail.sk   |               | 1.1.0001 0:00:00   | ✕ 🔑   |
| 5  | zamestnanec3  | zamestnanec3 | zamestnanec3 | zamestnanec3@mail.sk   |               | 1.1.0001 0:00:00   | ✕ 🔑   |
| 6  | zamestnanec4  | zamestnanec4 | zamestnanec4 | zamestnanec4@mail.sk   |               | 1.1.0001 0:00:00   | ✕ 🔑   |

## Menus.aspx – Jedálne lístky

## Jedálne lístky ?

## Kalendár

| máj 2013 |    |    |    |    |    |    |
|----------|----|----|----|----|----|----|
| ≤        |    |    |    |    |    | ≥  |
| po       | ut | st | št | pi | so | ne |
| 29       | 30 | 1  | 2  | 3  | 4  | 5  |
| 6        | 7  | 8  | 9  | 10 | 11 | 12 |
| 13       | 14 | 15 | 16 | 17 | 18 | 19 |
| 20       | 21 | 22 | 23 | 24 | 25 | 26 |
| 27       | 28 | 29 | 30 | 31 | 1  | 2  |
| 3        | 4  | 5  | 6  | 7  | 8  | 9  |

## Jedálny lístok ku dňu 24.5.2013

✕ Odobrať

| Označiť                     | Názov           | Typ          | Cena |
|-----------------------------|-----------------|--------------|------|
| <input type="checkbox"/> 3  | Bravčová roláda | Hlavné jedlo | 1€   |
| <input type="checkbox"/> 10 | Cesnaková       | Polievka     | 1€   |
| <input type="checkbox"/> 4  | Hovädzí tokaň   | Hlavné jedlo | 1€   |

## Obedy



+ Pridať

| Označiť                     | Názov              | Typ          | Cena |
|-----------------------------|--------------------|--------------|------|
| <input type="checkbox"/> 9  | Hrástková polievka | Polievka     | 1€   |
| <input type="checkbox"/> 1  | Pečené bravčové    | Hlavné jedlo | 1€   |
| <input type="checkbox"/> 5  | SETA DE POLLO      | Hlavné jedlo | 1€   |
| <input type="checkbox"/> 7  | Špenátová          | Polievka     | 1€   |
| <input type="checkbox"/> 31 | TestMeal           | Hlavné jedlo | 1€   |
| <input type="checkbox"/> 6  | Údený syr          | Hlavné jedlo | 1€   |
| <input type="checkbox"/> 2  | Zemiaková          | Hlavné jedlo | 1€   |
| <input type="checkbox"/> 8  | Zemiaková          | Polievka     | 1€   |

## History.aspx – História



OBEDÁR

Vítajte, zamestnanec0 ! 
[Domov](#)
[Objednávky](#)
[História](#)

## História objednávok

Časové rozmedzie:  – 


| Dátum     | Jedlo           | Stav    | Cena  |
|-----------|-----------------|---------|-------|
| 14.5.2013 | Pečené bravčové | Aktívna | 1,00€ |
| 9.5.2013  | Zemiaková       | Aktívna | 1,00€ |
| 9.5.2013  | Bravčová roláda | Aktívna | 1,00€ |