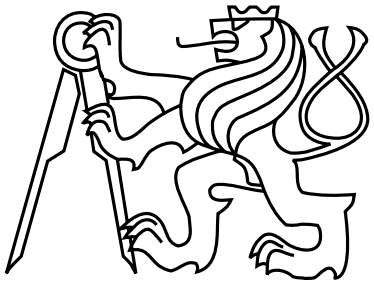
České vysoké učení technické v Praze
Fakulta elektrotechnická
Katedra počítačové grafiky a interakce



Bakalářská práce

Aplikace mobilní číšník pro restaurace na platformě OS Android

Václav Strnad

Vedoucí práce: Ing. Pavel Vít

Studijní program: Softwarové technologie a management, Bakalářský

Obor: Web a multimedia

23. května 2012

Poděkování

Děkuji vedoucímu práce Ing. Pavlu Vítovi za kontrolu a pomoc na projektu.

Prohlášení

Prohlašuji, že jsem práci vypracoval samostatně a použil jsem pouze podklady uvedené v přiloženém seznamu.
Nemám závažný důvod proti užití tohoto školního díla ve smyslu §60 Zákona č. 121/2000 Sb., o právu autorském, o právech souvisejících s právem autorským a o změně některých zákonů (autorský zákon).

V Praze dne 25. 5. 2012 .....

Abstract

This bachelors thesis is focused on application development for mobile OS Android. The aim of this application is to simplify the work of waiters, and thus increase the efficiency of the enterprise. The application will work so that a waiter receives the order from the guests, he writes it to the system and order will be sent to the central database, where other kitchen workers will be able to access it through the web application. The whole system is designed to simplify and accelerate the sharing of information between company employees. Acceleration of work with orders is in terms of efficient operation of the restaurant important to reduce the time customers wait for the ordered food. This step can increase the number of restaurant guests, thus turnover.

Abstrakt

Bakalářká práce je zaměřena na vývoj aplikace pro mobilní systém OS Android. Cílem této aplikace je zjednodušit práci číšníkům a tímto zvýšit efektivitu daného podniku. Aplikace bude pracovat tak, že číšník příjme objednávku od hostů, zapíše ji do systému a objednávka bude odeslána do centrální databáze, kde k ní ostatní pracovníci kuchyně budou mít přístup pomocí webové aplikace. Celý systém má za úkol zjednodušit a zrychlit sdílení informací mezi zaměstnanci podniku. Zrychlení práce s objednávkami je z hlediska efektivního provozu restaurace důležité pro to, aby se zákazníkům zkrátila doba čekání na objednané jídlo. Tímto krokem může restaurace zvýšit počet hostů, tudíž i obrat.

Obsah

1 Úvod **1**

2 Popis problému, specifikace cíle **3**
2.1 Specifikace cíle . . . 3
2.1.1 Funkční požadavky . . . 3
2.1.2 Vlastnosti výsledné práce . . . 3
2.2 Rešerše . . . 4
2.2.1 Konkurenční řešení . . . 4
2.2.1.1 Pokladní systém Harsys 6 . . . 4
2.2.1.2 Programovatelný systém CONTO . . . 5

3 Analýza a návrh řešení **7**
3.1 Analýza možných dílčích řešení . . . 7
3.2 Analýza ceny a spotřeby návrhu . . . 7
3.3 Časový plán . . . 8
3.4 Požadavky na výslednou platformu . . . 8
3.5 Analýza úzkých míst návrhu . . . 8
3.6 Stavební prvky řešení . . . 8
3.7 BPM . . . 9
3.8 Use cases . . . 9
3.9 Data Model . . . 11

4 Prototyp **13**

5 Realizace **17**
5.1 Webová aplikace . . . 17
5.1.1 Teorie . . . 17
5.1.2 Implementace . . . 18
5.2 Mobilní aplikace . . . 19
5.2.1 Teorie . . . 20
5.2.2 Implementace . . . 22
5.3 Další vývoj systému . . . 25

6 Testování **27**
6.1 Testování prototypu . . . 27
6.2 Testování grafického rozhraní . . . 28

7 Závěr **33**

Seznam obrázků

2.1 Pokladní terminál C.S.I. Star Touch 4
2.2 Pokladní terminál C.S.I. Star Touch 5
2.3 Vlevo: Síťová pokladní tiskárna Tysso PRP-085; Vpravo: Mobilní zařízení pro číšníkyC.S.I. French Touch 6

3.1 Vlevo: Proces objednání položky před nasazením systému; Vpravo: Proces objednání položky po nasazením systému 9
3.2 Vlevo: Proces zaplacení účtu před nasazení systému; Vpravo: Proces zaplacení účtu po nasazení systému 10
3.3 Případy užití v mobilní aplikaci 11
3.4 Tabulky v databázi 12

4.1 Přihlášení do systému 13
4.2 Vlevo: Výpis účtenek v systému; Vpravo: Podrobnosti účtenky. 14
4.3 Vlevo: Kategorie jídelního lístku; Uprostřed: Výpis specifické kategorie jídelního lístku; Vpravo: Podrobnosti položky jídelního lístku. 14
4.4 Vlevo: Kategorie nápojového lístku; Uprostřed: Výpis specifické kategorie nápojového lístku; Vpravo: Podrobnosti položky nápojového lístku. 15

5.1 Životní cyklus presenteru v Nette Framework, převzato z Nette 18
5.2 Vlevo: Vývojové prostředí Eclipse; Vpravo: Emulované prostředí pro test aplikací 20
5.3 Životní cyklus activity, převzato z Android Developers 21

6.1 Sony Ericsson Xperia mini pro 28
6.2 Druhý pokus o přihlášen 30
6.3 Pohyb v seznam 30
6.4 Vlevo: Podrobnosti stolu; Vpravo: Seznam s tlačítkem „zpět“ 31
6.5 Zobrazení Toastu 31

Kapitola 1

Úvod

Snahou provozovatelů restauračních zařízení je obsloužit co nejvíce hostů v co nejkratší době, aniž by došlo ke zhoršení kvality jídla a obsluhy. Vzhledem k tomu, že doba přípravy jídel je dána vybavením kuchyně a zkušeností a dovedností kuchaře, zaměříme se na zefektivnění práce číšníků. Cílem bakalářské práce je vytvořit aplikaci, která jim pomůže zkrátit dobu od přijetí objednávky do začátku vaření, zjednoduší a zrychlí komunikaci číšníka s kuchařem, připraví podklady pro snadné vytvoření účtenky, a která v neposlední řadě poskytne majiteli jasný přehled o provozu jeho restaurace. Snadno a rychle tak může reagovat na případné nedostatky chodu zařízení. Tento nový systém mu poskytne jasné podklady pro hodnocení kvality práce personálu. Aplikace nahradí často používaný "papírkový"systém přijímání objednávek, a tak i odstraní chyby, které vznikají jeho používáním.

„Papírkový“ systém spočívá v tom, že číšník s bločkem v rychlosti napíše objednávku, kterou poté odnáší do kuchyně a tlumočí nebo přepisuje ji kuchaři. Tento systém je pomalý a mohou v něm lehce vznikat chyby. Aplikace tedy tyto zbytečné kroky přeskočí a nabídne jasné informace o objednávkách.

Aplikace nabídne číšníkům dostatečně jednoduché menu pro rychlý přesun z elektronického jídelního lístku do objednávky, ty bude aplikace automaticky vkládat do centrální databáze, která bude přístupná pouze zaměstnancům restaurace. Objednávky budou obsahovat informaci o tom, kdo si co objednal a kdo a kdy objednávku přijal.

Systém tedy bude obsahovat tři uživatelské role, číšník, majitel a kuchař. Každá z těchto rolí vyžaduje jiný druh přístupu i správy. Číšník bude využívat rychlý přístup z mobilní aplikace. Majitel využije přehledný výpis a správu všech informací v systému. Kuchař bude potřebovat jasný výpis objednávek, tak aby neztrácel čas s hledáním a čtením objednávek.

Kapitola 2

Popis problému, specifikace cíle

2.1 Specifikace cíle

2.1.1 Funkční požadavky

Správa uživatelů Majitel restaurace bude moci přidávat a odstraňovat uživatele v systému.

Správa jídelního a nápojového lístku V jídelním a nápojovém lísku budou uživatelé aplikace moci přidávat kategie lístku, měnit je a odstraňovat. Do vzniklých kategorií budou poté moci přidávat, upravovat a odstraňovat jednotlivé položky.

Správa účtů Číšník bude oprávněn vytvářet účty pro nově příchozí hosty. Účet bude mít možnost být přesunut k jinému stolu. Nakonec bude čísník moci účet ukončit.

Správa objednávek Čísník bude moci přidat novou objednávku na vybraný účet. V případě chyby objednávku bude moci změnit nebo rovnou odstranit. Po dokončení přípravy objednávky k servírování označí kuchař objednávku za připravenou.

2.1.2 Vlastnosti výsledné práce

Uživatelé Cílovou skupinou uživatelů jsou číšníci, kterým aplikace umožní rychlé a jednoduché zadávání objednávky do systému restaurace. Zjednoduší se opravy případných chyb při objednávkách a urychlí číšníkům komunikaci s ostatními zaměstnanci restaurace.

Aktivity Číšník při příchodu k zákazníkům vybere stůl, ke kterému v systému vytvoří nový účet. Hosté nadiktují objednávku číšníkovi, který v seznamu vybere jednotlivé položky, čímž je přidá na daný účet. Pokud hosté budou chtít změnit objednávku, číšník opraví položky podle přání. Ve chvíli, kdy bude objednávka v pořádku, číšník ji odešle objednávku do systému. V tuto chvíli si objednávku už převezme kuchař, popřípadě barman. Až bude objednávka připravena, dostane odpovědný číšník zprávu, aby odnesl objednávku zákazníkům. Nakonec číšník uzavře účet, nechá ho vytisknout a doklad o zaplacení předá zákazníkovi.

Uzavřením účtu se v systému stůl uvolní a bude možné po jeho obsazení dalším zákazníkem vytvořit nový účet. Tak začne cyklus znova.

Podpora Vytvoření proběhne přímo na stůl, který je volný. Objednávka bude definovaná účtem a číšníkem, který ji vytvořil, tudíž až bude objednávka hotová, daný číšník obdrží upomínku aby odnesl objednávku.

Aplikace bude vytvořena pro použití na mobilním zařízení se systémem Android. Databáze, ke které bude aplikace přistupovat bude na PC připojené pomocí Wi-fi sítě.

Kontext Nejdůležitější vlastností aplikace bude jednoduchost. Čísník bude muset v podstatě zároveň vidět položky, které si budou hosté vybírat, a aktuální účet pro případnou korekci. Dále musí být možný rychlýpřesun, tak aby se objednávky daly rychle zadávat.

2.2 Rešerše

2.2.1 Konkurenční řešení

2.2.1.1 Pokladní systém Harsys 6

[2]

Popis: Restaurační software od firmy ABX software s.r.o. . Jedná se o celkový návrh systému, který umožňuje správu skladu, zaměstnanců, jídelního lístku včetně receptur a další. Požadovaný operační systém, ve kterém bude software nainstalovaný, je Windows 2000 a novější. Data se Ukládají v databázi SQL Firebird. Používaný hardware je například dotykový terminál (Obrázek 2.1), dále server s uloženými daty a pokladní tiskárna. Možnost rozšíření tohoto systému, je použít tzv. Mobilní PDA číšník. Toto PDA, po připojení do systému pomocí sítě Wi-fi, jako standartní terminál.

Obrázek 2.1: Pokladní terminál C.S.I. Star Touch

Možné způsoby instalace systému:

- Systém lze instalovat na jediné zařízení, které slouží jednak jako pokladna, jednak i jako úložiště dat.
- Systém lze instalovat na více zařízení, která slouží jako pokladny, jako počítače v kancelářích nebo jako server. Veškeré propojení je realizováno prostřednictvím interních sítí.
- Systém lze instalovat na různých zařízeních jako v druhém bodě. Propojení je realizované přes internet.
- Dalším způsobem instalace je propojení různých zařízení v síti, prostřednictvím které si tato zařízení informace předávají. Zařízení jsou po synchronizaci soběstačná.

Výhody systému: Pokladní systém Harsys 6 díky své dlouhé existenci obsahuje mnoho nástrojů pro správu skladu a provoz restaurace. Nabízí podporu různých periferií a pokladen.

Nevýhody systému: Tento software je vytvořený pouze pro systém Windows. Technika používaná v sestavách má vysokou pořizovací cenu.

2.2.1.2 Programovatelný systém CONTO

[7]

Popis: Modulární systém pro hotely, obchody a restaurace firmy Novotný – Atrima Brno. Software využívá relační databázi Firebird SQL, podporuje formát XML pro sdílení informací. Výhodou systému je správa účtů v podobě graficky znázorněné mapy stolů. Návrh využívá dotykový terminál (Obrázek 2.2) připojený k serveru a síťové tiskárně (Obrázek 2.3 Vlevo) pro vytištění účtenek. Rozšířením je mobilní zařízení (Obrázek 2.3 Vpravo), které nabízí přístup do systému odkudkoli. Jediné omezení je dosah místní sítě v které je připojené.

Obrázek 2.2: Pokladní terminál C.S.I. Star Touch

Výhody systému: Velkou výhodou systému Conto je možnost jeho úpravy pro individuální potřeby restauračních zařízení. Použití tohoto systému je výhodné především pro zařízení, které nabízejí kombinaci restauračních, hotelových zařízeních i obchodech.

Nevýhody systému: Modulárnost systému je zároveň jeho nevýhodou. Uživatelské rozhraní totiž obsahuje nadmíru informací, které snižují schopnost orientace v systému. Další nevýhodou je velikost použitého mobilního zařízení. Ta navíc mohou být v systému připojena pouze čtyři.

Používaný hardware se systémem CONTO

Aplikace na zařízení PDA, která slouží jako rozšíření pokladního systému. Aplikace simuluje pokladnu v mobilním zařízení.

Obrázek 2.3: Vlevo: Síťová pokladní tiskárna Tysso PRP-085; Vpravo: Mobilní zařízení pro číšníkyC.S.I. French Touch

Kapitola 3

Analýza a návrh řešení

3.1 Analýza možných dílčích řešení

Systém se zaměřuje na správu dat v restauraci. Pro takovýto účel se nejvíce hodí informace ukládat databázovým systémem, který dokáže pojmout obrovské množství dat ve strukturované podobě. Proto, aby byly náklady na software co nejnižší, využijeme open source databázi, možnostmi je SQLite nebo MySQL. SQLite se většinou využívá ve spojení s aplikacemi implementované v Javě. Ve webových databázích se zase používá MySQL databáze. Tyto systémy mohou být nasazeny ve dvou způsobech, prvním je instalace na placeném serveru, druhou je soukromé zařízení s instalovaným webovým serverem.

Centrální aplikaci je potřebné implementovat v jazyce, který bude multiplatformní a bude jednoduché jej nasadit. Možnostmi jsou Java nebo PHP. Oba jazyky umí pracovat s databázemi a jejich nasazení není problémem. V obou případech je potředa nainstalovat prostředí, v kterém poběží. V případě Javy se jedná o JRE. U PHP jde o stejnojmený program v kombinaci se serverem Apache.

Mobilní aplikace musí k centrální aplikaci přistupovat přes šifrované spojení, aby zamezilo úniku dat při bezdrátové komunikaci. Požadavkem bylo určeno, že aplikace je určena pro OS Android, což je Java obohacená o knihovny připravené k práci s periferiemi mobilních zařízení.

3.2 Analýza ceny a spotřeby návrhu

Pořizovací cena hardwaru dostatečně výkonného pro běh jednoho serveru s databází a aplikací je 10000 Kč. Pokud bychom chtěli použít dotykový terminál s instalovaným serverem, jeho cena bude ještě vyšší. V případě konkurenčních řešení se jedná částky okolo 25000 Kč.Jediným omezením systému je nutnost použití mobilního zařízení s OS Android nejlépe s dotykovou obrazovkou, v kterém lze využít intuitivní ovládání. Ceny těchto zařízení se pohybují již od 3000 Kč. Cena softwaru není určena. Ceny jsou pouze orientační, získané z internetového obchodu Alza [3], ke dni 8. 5. 2012.

3.3 Časový plán

Vývoj systému je součástí bakalářské práce. Proto musí být zprovoznění základní funkcionality hotové ke dni 25. 5. 2012. Implementace dalších funkcí systému a technologií bude probíhat během následujících dvou měsíců tak, aby bylo možné systém plně nasadit v provozu restauračního zařízení.

3.4 Požadavky na výslednou platformu

Nízká cena Software nebude využívat placený software třetích stran. Cílem je dosáhnout co nejnižší ceny.

Ovladatelnost aplikace Aplikace musí mít rychle ovladatelné rozhraní. Nasazení bude v nejrůznějších prostředích, zatížených např. hlukem i nedostatkem světla.

Systém mobilní aplikace Mobilní aplikace bude navržena pro systém OS Android. Výhodou zařízení s tímto systémem je jejich rychlé rozšiřování, způsobené nabídkou kvalitních služeb a hardwarem ve všech cenových kategoriích.

3.5 Analýza úzkých míst návrhu

Přerušení spojení mobilní aplikace se serverem Aplikace bude navržena tak, aby byla schopná samostatného fungování v případě přerušení spojení s centrální databází.

Neoprávněný přístup Webová aplikace musí být navržena tak, aby osoba bez přístupových údajů nemohla získat ani upravovat data v systému.

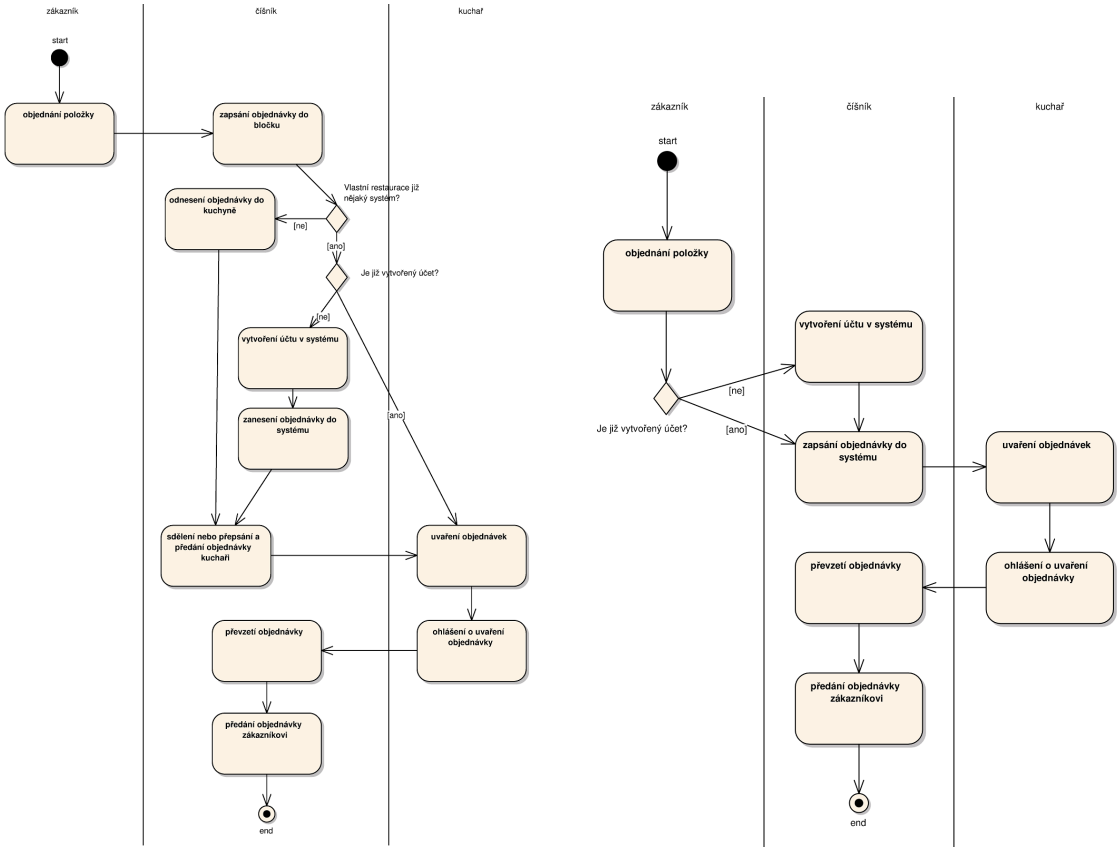
Konflikty nahrávaných dat Databáze centrální aplikace bude získávat data od mobilních aplikací. Aplikace tedy budou posílat pouze informace o tom, jak data změnit, ne však výsledný stav vzniklý touto změnou.

3.6 Stavební prvky řešení

Celý systém bude tvořen alespoň jedním mobilním zařízením a serverem, ke kterému se mobilní aplikace připojí v případě nedostatku vlastních informací nebo za účelem vložení nových dat. Data budou uložena na databázovém serveru MySQL instalovaném podle požadavků zákazníka, na hostingovaném serveru nebo na osobním počítači, připojeném k síti Wi-fi. Serverová část bude mít webový přístup pro správu uživatelům. Mobilní aplikace bude navržena pro použití na zařízení s operačním systémem OS Android. Kvůli požadavku na rychlé ovládání, bude řešení obsahovat zařízení se systémem ve verzi 3.0 a vyšší. Důvodem jsou propracovanější ovládací prvky oproti starším verzím.

3.7 BPM

Restaurační zařízení je místem, ve kterém jsou zavedeny určité procesy, které je těžké měnit. Nový systém proto musí tyto zažité postupy téměř kopírovat. Mobilní aplikace by tedy měla pokrývat veškeré procesy spojené s objenávkami v restauracích. Proces zadání objednávky probíhá dnes takto (Obrázek 3.1 Vlevo), po nasazení systému bude probíhat takto (Obrázek 3.1 Vlevo). Tímto se dosáhne většího využití času číšníků. Číšníci budou moci být více u hostů, což předpokládám povede k jejich větší spokojenosti. Další možností jak zefek-

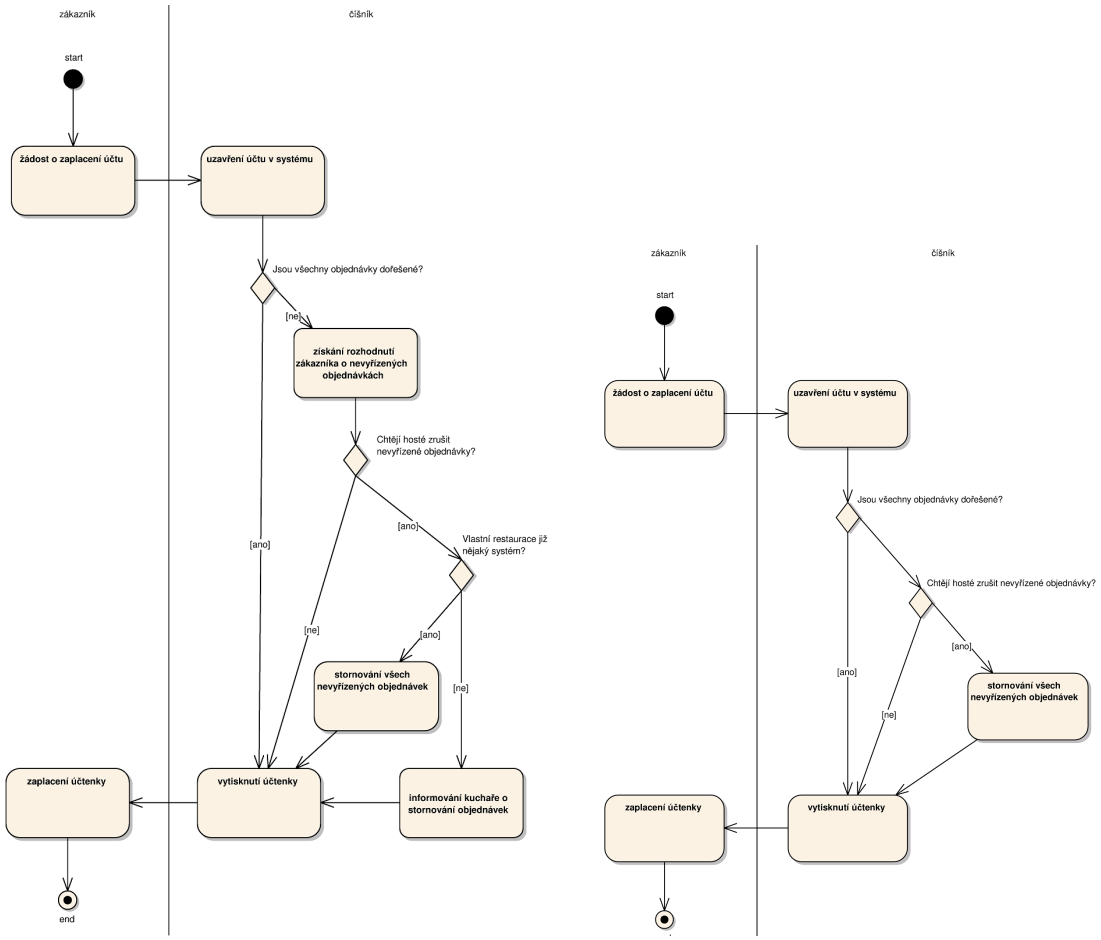


Obrázek 3.1: Vlevo: Proces objednání položky před nasazením systému; Vpravo: Proces objednání položky po nasazením systému

tivnit chod restaurace je zaplacení účtu. Stav před nasazením (Obrázek 3.2 Vlevo) a stav po nasazení systému (Obrázek 3.2 Vpravo).

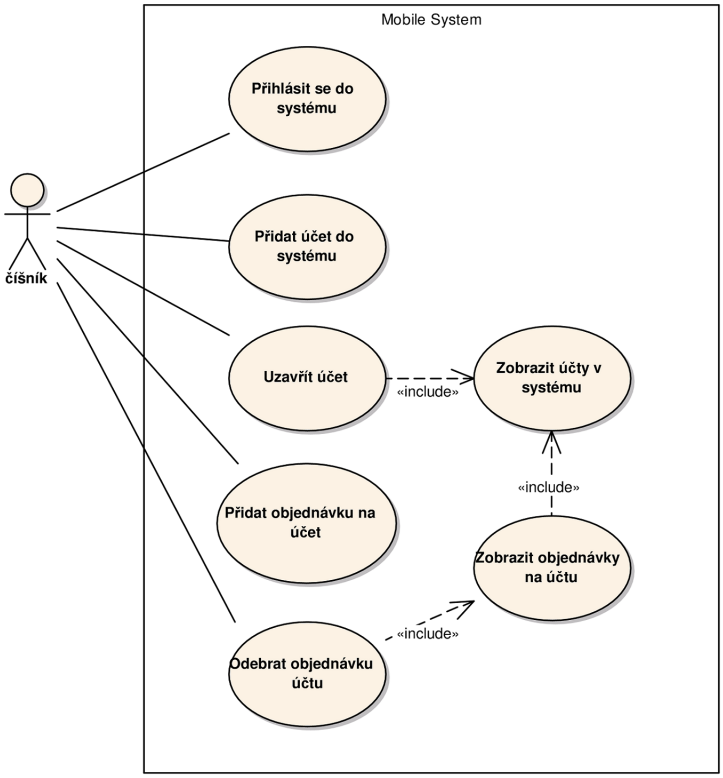
3.8 Use cases

Podle požadavků jsem vytvořil případy užití, které je všechny musí pokrýt. Podle případů se poté naimplementují jednotlivé části systému. Nejdůležitější částí je mobilní aplikace,



Obrázek 3.2: Vlevo: Proces zaplacení účtu před nasazení systému; Vpravo: Proces zaplacení účtu po nasazení systému

která má následující případy (Obrázek 3.3).



Obrázek 3.3: Případy užití v mobilní aplikaci

3.9 Data Model

Server bude všechna data ukládat do své MySQL databáze. Pro účely objednávek budou muset být vytvořeny následující tabulky. (Obrázek 3.4)

Tabulka `users` s daty o uživatelích systému. Tabulka obsahuje id záznamu, uživatelské jméno, email, křestní jméno a příjmení a roli, kterou zastupuje v systému. Pro přihlášení je uloženo heslo a takzvaná sůl pro zvýšení počtu kombinací hesla, čímž bude zajištěna větší bezpečnost systému.

Tabulka `tables` obsahuje záznam pro každý stůl v restauraci. U každého záznamu evidujeme id, jméno a tag uchovávající označení stolu.

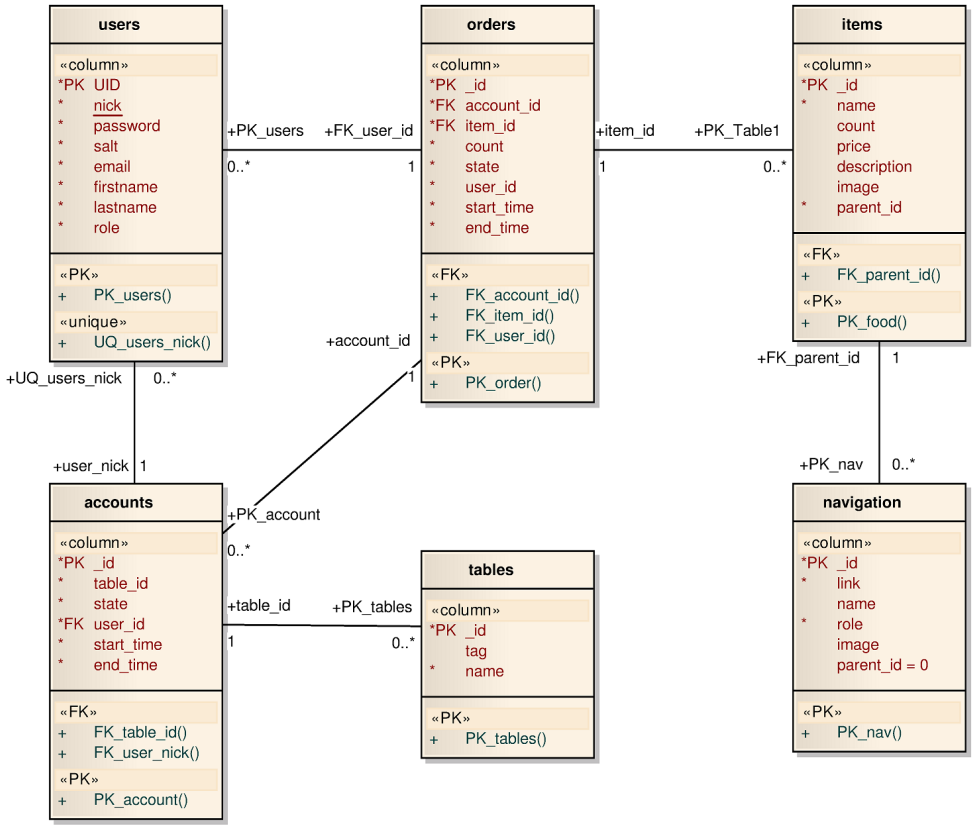
Tabulka `accounts` ukládá záznamy o zákaznických účtech. Účty obsahují id, cizí klíč s id stolu u kterého je účet vytvořený, stav, ve kterém se účet nachází, cizí klíč s id číšníka, který účet vytvořil, čas vytvoření účtu a čas, kdy si zákazník přál zaplatit.

Tabulka `navigation` obsahuje navigaci we webovém menu a kategorie v jídelním a nápojovém lístku. Opět zde najdeme id záznamu, link určující o jaký presenter jde ve webové aplikaci, popisek položky, role které mají do této sekce přístup, pozici uložení ilustračního obrázku a id rodiče. V případě kořenového prvku je uložená nula.

Tabulka `items` ukládá všechny položky jídelního lístku v databázi. Jde zároveň i o jakýsi seznam skladu. Obsahuje totiž id položky, popisek, množství porcí, které si hosté ještě mohou objednat. Dále obsahuje sloupce s cenou položky, podrobný popis položky, pozici uloženého

obrázku a cizí klíč s id kategorie v tabulce `navigation`, kam položka patří.

Nejrozsáhlejší tabulkou je tabulka `orders`. Kromě id obsahuje cizí klíč s id účtu, na který se má připsat, cizí klíč s id položky, kterou si host objednal, množství označující kolik si host objednal položek, stav objednávky, cizí klíč s id na uživatele systému, který objednávku vytvořil, čas vytvoření objednávky a čas vyřízení objednávky.



Obrázek 3.4: Tabulky v databázi

Kapitola 4

Prototyp

[4]

Prototyp uživatelského rozhraní je vytvořený v aplikaci Balsamiq Mockups, kterou je možné používat na webových stránkach Balsamiq nebo ji instalovat na PC. Tato aplikace umožňuje navrhnout rozhraní aplikace pro otestování dříve, než se začne vytvářet samotný program. Designové rozdíly mezi systémy iOS a OS Android nejsou při testování uživatelského rozhraní podstatné, intuitivnost ovládání je srovnatelná. Celý prototyp slouží pouze k otestování grafického rozhraní. Testováním takového prototypu můžeme zjistit nedokonalosti rozhraní ještě před započetím implementace.

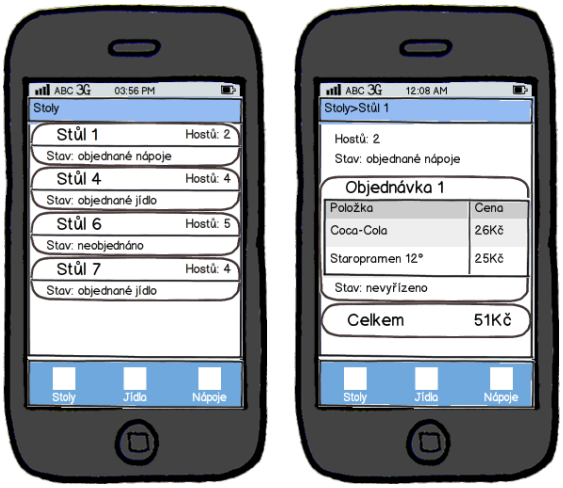
V aplikaci Balsamiq Mockups jsem navrhl rozhraní prezentující číšníkům v restauraci otevřené účty, jídelní a nápojový lístek. Prototyp obsahuje tři typy zobrazení. Jedním je obrazovka pro přihlášení, další je výpis seznamu a poslední vypisuje podrobnosti určité položky.

První typ, který uživatel uvidí, je přihlašovací obrazovka pro autentizaci uživatele před vstupem do systému (Obrázek 4.1). Po přihlášení uvidí uživatel na obrazc tři tlačítka, která simulují záložky oddělující od sebe účtenky, jídelní a nápojový lístek. Pod první

Obrázek 4.1: Přihlášení do systému

záložkou se nachází výpis účtenek (Obrázek 4.2 Vlevo). Po kliknutí na jednu z účtenek se uživateli zobrazí podrobnosti zvolené účtenky (Obrázek 4.2 Vpravo).

Pod druhou záložkou se nachází jídelní lístek. Zde se uživateli zobrazí kategorie jídel



Obrázek 4.2: Vlevo: Výpis účtenek v systému; Vpravo: Podrobnosti účtenky.

(Obrázek 4.3 Vlevo). Po výběru kategorie vidí číšník seznam položek společně s jejich základními informacemi (Obrázek 4.3 Uprostřed). Po kliknutí na zvolenou položku se zobrazí třetí typ obrazovky, podrobný výpis (Obrázek 4.3 Vpravo). Pokud si zákazník bude chtít objednat danou položku, číšník klikne na tlačítko objednat. V případě, že si položku bude chtít objednat více zákazníků, číšník vybere z nabídky odpovídající množství.

Pod třetí záložkou se nachází nápojový lístek. Obsahuje kategorie nápojů (Obrázek 4.4

Obrázek 4.3: Vlevo: Kategorie jídelního lístku; Uprostřed: Výpis specifické kategorie jídelního lístku; Vpravo: Podrobnosti položky jídelního lístku.

Uprostřed). Jednotlivé kategorie obsahují příslušné položky nápojového lístku(Obrázek 4.4 Uprostřed). Každou položkusi stejně jako u jídelního lístku může uživatel nechat vypsat podrobně (Obrázek 4.4 Vpravo). Zde vybírá počet položek, které si hosté objednávají.

Prototyp uživatelského rozhraní se skládá ze základních ovládacích prvků. Díky kom-

Obrázek 4.4: Vlevo: Kategorie nápojového lístku; Uprostřed: Výpis specifické kategorie nápojového lístku; Vpravo: Podrobnosti položky nápojového lístku.

binacím desítek prvků lze vytvořit téměř jakýkoli návrh tak, aby vyhovoval požadavkům výsledné aplikace. Výsledný prototyp může obsahovat pouze jednotlivé průchody, které budou použity při testování. Díky aplikaci Balsamiq Mockups lze některým prvkům vytvořit odkaz k jiné obrazovce. Participant poté v prototypu označuje ovládací prvky, které měl problém najít, nebo prvky, které hledal a v prototypu vůbec nebyly.

Prototyp je vytvořený pouze pro otestování ovladatelnosti v hektickém prostředí. Číšník musí mít možnost okamžitého přesunu mezi jednotlivými položkami jídelního a nápojového lístku. Zároveň musí mít možnost rychlé opravy v případě, že host nebo samotný číšník při objednávání udělá chybu.

Kapitola 5

Realizace

Systém je rozdělen do dvou implementačních částí. Jednou z částí je důvod projektu, mobilní aplikace, která bude pomáhat číšníkům s objednávkami. Druhou částí je podpora aplikace a možnost konfortnější správy databáze, webová aplikace.

5.1 Webová aplikace

Webová aplikace slouží pouze jako správcovský a prezentační nástroj, tudíž jsem se rozhodl využít framework, který velice zjednodušší práci s databází a prezenční vrstvou v systému. Framework Nette jsem si vybral hlavně kvůli velké základně programátorů, kteří již mnohokrát řešili problémy, s kterými jsem se později setkal při vývoji.

5.1.1 Teorie

[6]

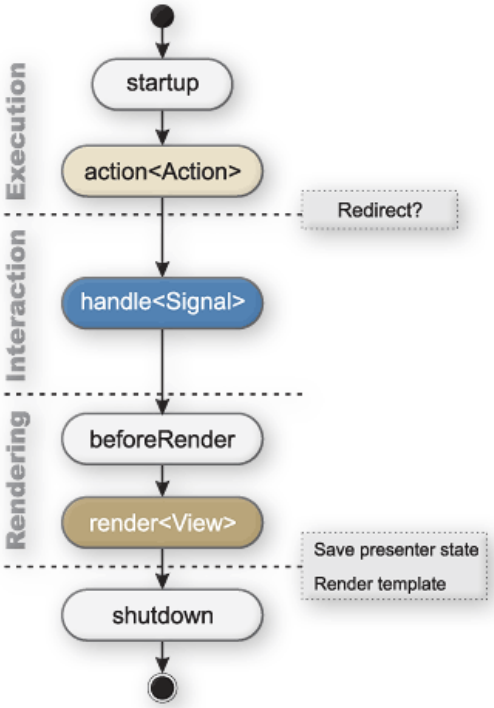
Hlavní složkou projektu, obsahující celou funkcionalitu aplikace, je složka `app`. Aby framework mohl efektivně spravovat databázi, musel jsem upravit soubor `./config/config.neon`. Ten obsahuje nastavení přístupu k databázi a dále tzv. „factories“, které vytvážejí v systému objekty, ty jednotlivé tabulky databáze a funkce, které nad ní můžeme volat. Třídy reprezentující objekty jsou uloženy ve složce `./model`, view aplikace je uloženo ve složce `./templates` a controler je zde reprezentovaný „presentery“ ve složce `./presenters`.

Příchozí požadavek na aplikaci parsuje URL podle vzoru v souboru `./app/bootstrap.php`. Odsud je přesměrovaný na vybraný presenter a šablonu. Dále v požadavku může být uložena cookie, kterou již presenter použije například pro autentizaci uživatele.

Životní cyklus presenteru

Každý presenter má svůj životní cyklus (Obrázek 5.1), během kterého se volají dané funkce. První funkcí je `startup()`, v které se provádí obecné operace jako je autentizace, kontroly a případné přesměrování. Zatím nadochází k práci s daty specifickými pro daný presenter. Ve funkci `action()` provádíme operace, které ovlivňují následující render funkci, jako je výběr použité šablona pro vykreslení atp. Ve funkci handle jsou odchytávány signály například

odeslání formuláře s daty. Následuje funkce `beforeRender()`, kde si předpřipravíme data pro vykreslení. Předem připravená data využijeme ve funkci `renderNázev()`. Název je složen ze dvou částí, „render“ značí, že se jedná o funkci pracující s šablonou a druhá část je názvem šablony, která se použíje.



Obrázek 5.1: Životní cyklus presenteru v Nette Framework, převzato z Nette

Šablony

V Nette se jako šablonovací jazyk využíváme jazyk „latte“. Každý presenter má alespoň jednu svou vlastní šablonu, pokud se nespecifikuje jinak, volá se šablona `default.latte`. Pokud je šablon více, vyběr té, která se nakonec bude vykreslovat, provedeme ve funkci action. Například může dojíd k přesměrování na chybové hlášení. Všechny tyto templaty se při vykreslování, postupně vkládají do základní šablony `@layout.latte`, který definuje hlavičku dokumentu a jeho základní strukturu.

5.1.2 Implementace

Aplikace obsahuje 6 ruzných typů, je jím objekt pro účtenky (Accounts), položky v jídelním a nápojovém lístku (Items), kategorie menu a navigace (Navigation), objednávky (Orders), stoly (Tables) a uživatele (Users).

Implementované presentery pro správu

Prvním presenterem při vstupu na stránky, je `HomepagePresenter.php`. Dále složka obsahuje `SignPresenter.php`, na který jsme přesměrovaní, pokud nejsme přihlášeni. Hlavním presenterem, od kterého většina dědí, je `SecurePresenter.php`, ten všem potomkům definuje funkci `startup()`, kde se zjistí, jestli je uživatel přihlášený a pokud není, je přesměrován na již zmiňovaný `SignPresenter.php`. Dalším obecným presenterem je `ErrorPresenter`, na který se přesměruje, pokud se vyskytne v aplikaci chyba jako požadavek na presenter, který v aplikaci neexistuje.

Následují presentery starající se o správu obsahu `AccountsPresenter`, `ItemsPresenter`, `OrdersPresenter`, `UsersPresenter` a presenter `UserPresenter`, umožňující změnu osobních údajů. Každý z těchto presenterů obsahuje výpis položek, které do danného presenteru patří a formuláře pro přidání a úpravu danných položek. Vytvoření formuláře se provádí ve funkcích s dannou strukturou hlavičky. Parametr `name` označuje název formuláře, který je jmenován v šabloně. Tím je možné rozlišit více formulářů na stránce.

```
protected function createComponentNázev($name){
...
}
```

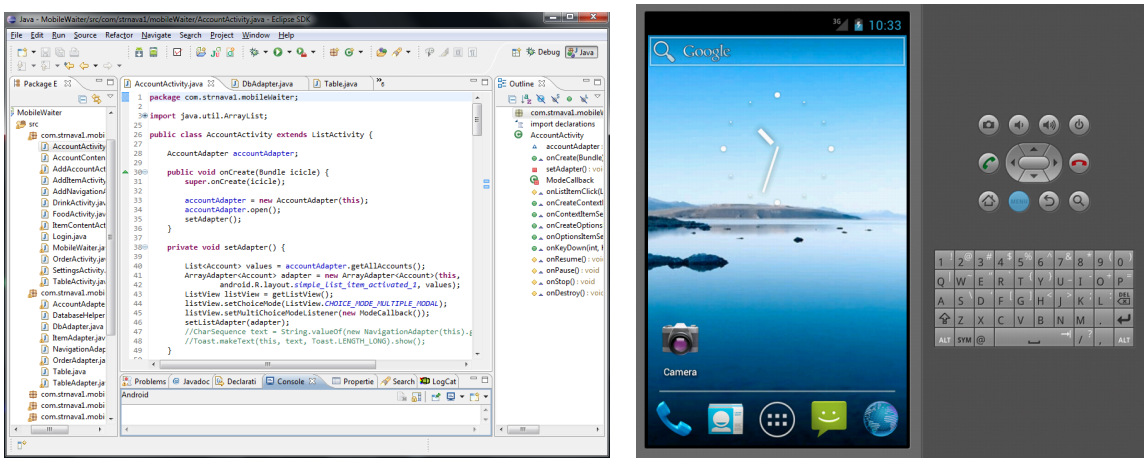
Ve funkci jsou poté vyjmenované položky, které se mají vykreslit, současně s vytvářením se i nastaví jejich typy a podmínky, které se mají hlídat. Standardně se formulář odesílá metodou POST. Při odeslání formuláře, se data odešlou funkci pro odchycení signálu, ta poté zpracuje a uloží data do databáze.

Presenter pro synchronizaci dat

Stěžejním presenterem je `JsonPresenter`, ten se stará o komunikaci s mobilní aplikací. Presenter si z požadavku získá POST data, která jsou označena jako „json“ a uloží si je v decodované podobě. Tyto data obsahují pokyny ke spuštění specifické funkce v dané třídě. Volání funkce je realizované univerzální metodou `call_user_method()`. Výsledek je zpětně převeden do syntaxe json a odeslán jako odpověď.

5.2 Mobilní aplikace

Aplikace je vyvíjena v programu Eclipse Indigo (Obrázek 5.2 Vlevo) s nainstalovanou knihovnou SDK v4.0. SDK v sobě kromě nápovědy obsahuje ještě programy, umožňující vzniklý kód okamžitě nahrát do zařízení s OS Android připojeného k počítači nebo vytvořit grafické virtuální prostředí, které simuluje celý mobilní telefon (Obrázek 5.2 Vpravo). Toto prostředí má ale oproti skutečnému mobilnímu zařízení velkou nevýhodu. Simulovat takovéto prostředí v počítači je výpočetně i paměťově náročné. Proto není zcela vhodné na něm testovat výkonnost nebo stabilitu aplikace.



Obrázek 5.2: Vlevo: Vývojové prostředí Eclipse; Vpravo: Emulované prostředí pro test aplikací

5.2.1 Teorie

Každá mobilní aplikace vytvořená pro OS Android musí obsahovat tzv. `AndroidManifest.xml`, umístěný v úvodní složce projektu, jde o XML soubor definující samotnou aplikaci. Obsahuje, pro jakou verzi SDK je kód tvořen a jaký minimální je vyžadován, jelikož aplikace využívá prvky z novějších verzí systému. Pokud aplikace bude pracovat s nějakou periferií zařízení, musí zde specifikovat oprávnění k nim. Jsou jimi například oprávnění k přístupu na internet, práce s vybracemi a další. Nakonec zde hlavně najdeme entity `activity`. Jde o záznamy všech activit v aplikaci, které se budou zobrazovat uživateli. Pokud nějaká z nich nebude zde zmíněná, aplikace activitu neotevře a zahlásí chybu. Každá znich obsahuje název třídy, v které je aktivita implementována. Pochopitelně entity neobsahují pouze atributy se jménem, ale i můžeme zde nastavit i popisek nebo základní styl, ve kterém se obrazovka vykreslí. Záznam hlavní activita, která se má spustit když kliknete na ikonu aplikace v seznamu aplikací, obsahuje `intent-filter`.

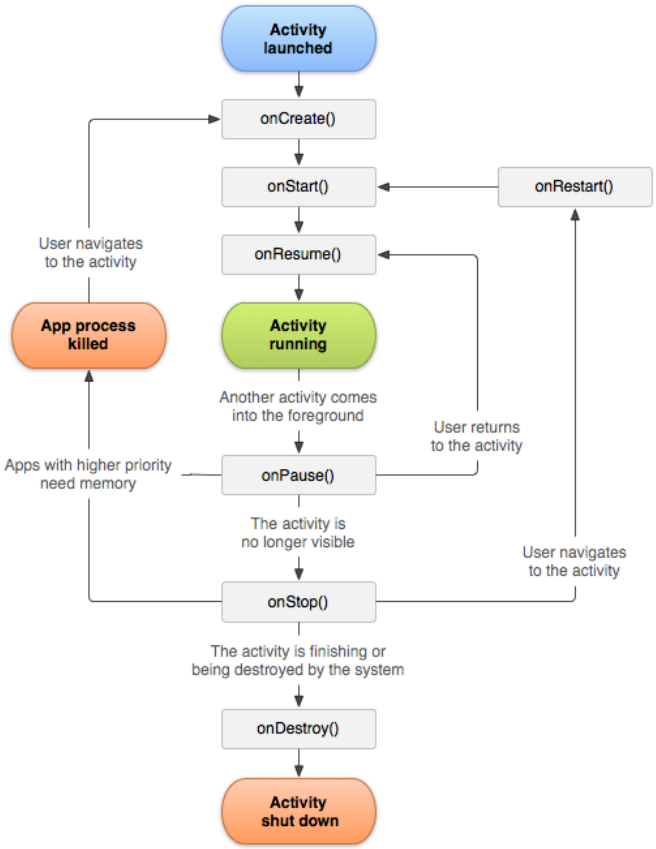
```
<intent-filter>
<action android:name="android.intent.action.MAIN" />
<category android:name="android.intent.category.LAUNCHER" />
</intent-filter>
```

Ten, jak název napovídá, filtruje označuje activitu za hlavní a spustitelnou. Případě přijetí odpovídající zprávy, kterou je příkaz ke spuštění aplikace, se aktivuje právě tato aktivita a ne jiná z aplikace.

Activity

Nyní přejdeme k již zmíněným activitám, uložených ve složce `./src`. Activita je třída, která ovládá grafické rozhraní. Stejně jako presentery v Nette mají životní cyklus, kterým se

prochází od chvíle, kdy activita vzniká až do doby, kdy zanikne (Obrázek 5.3). Metoda `onCreate()`, která se volá při spuštění activity, slouží k inicializacím proměnných a k prvotnímu vykreslení obrazovky. Naopak v metodě `onDestroy` uzavíráme komunikaci a activita končí.



Obrázek 5.3: Životní cyklus activity, převzato z Android Developers

Velkou částí aplikace jsou XML soubory definující rozličný obsah a obrázky uložené v adresáři `./res`. Rozložení a základní strukturu obrazovky, je definované v šablonách, v adresáři `./res/layout`. Tyto šablony vytváříme, pokud potřebujeme mít specializovanou strukturu obrazovky. Šablona obsahuje kořenovou entitu určující strukturu. Do té se vkládají další entity definující prvky na obrazovce. Elementy obsahují atributy id pro odlišení více stejných elementů na obrazovce, nastavení zarovnání nebo předdefinovaný text a další.

Menu

V aplikaci můžeme dále využít menu `./res/menu`. Kromě uložení od ostatních XML souborů se odlišují kořenovým elementem `<menu>` obsahující prvky `<item>`, což jsou již samotné položky menu. Atributy definují id a v praxi se jim už předem nastaví popisek.

Předdefinované hodnoty v aplikaci

Určité hodnoty, jako třeba texty, si můžeme předdefinovat v souborech v adresáři `./res/values`. Jde rovněž o XML dokumenty. Kořenovým elementem v těchto souborech je `<resources>`, ty obsahují prvky podle toho jakou hodnotu si chceme předefinovat. Příkladem element pro předdefinování si textu obsahující název aplikace.

```
<string name="app_name">MobileWaiter</string>
```

Obrázky

Ikony, které bychom chtěli někde vykreslovat, musíme uložit do složek začínajících `drawable`, ve formátu PNG a zároveň ve dvou formátech, jeden pro stisknutý a druhý pro nestisknutý stav. Jelikož systém OS Android je na zařízeních s různým rozlišením a formát PNG není vektorový, řeší se to uložením ikon rovněž v několika rozlišeních. Ikony s extrémně vysokým rozlišením 96 x 96 px se uloží do složky `drawable` s příponou `-xdpi`,s vysokým rozlišením 72 x 72 px do složky s příponou `-hdpi`, se střední kvalitou 48 x 48 do složky s příponou `-mdpi` a nakonec nejnižší kvalita 36 x36 px ve složce s příponou `-ldpi`. Verzi ikony si poté aplikace automaticky vybere ze souboru, kde jsou verze ikon s danným názvem zaindexované.

5.2.2 Implementace

[5] [1]

Po nainstalování SDK do programu Eclipse jsem mohl rovnou vytvořit „Android Project“. Eclipse mi poté vygeneroval celou strukturu se základními soubory včetně úvodní activity s názvem projektu MobileWaiter. Rovněž se teda i tato activita nastavila v AndroidManifestu jako hlavní a má se tedy spustit, pokud vyvolám aplikaci v menu.

Hlavní activita obsahuje definici záložek, proto jsme ji nastavili jako potomka třídy TabActivity, tedy activity, která bude spravovat záložky. Každou ze záložek musíme vytvořit zvlášť. Nejdříve si vytvoříme jakýsi spouštěč, kterému nastavíme třídu, kterou chceme, aby se otevřela pod záložkou, pokud na ni klikneme. Dále získáme view, do kterého budeme ukládat záložky a uložíme do proměnné `tabhost`. Samotné naprogramování záložky přichází teprve teď. Nad objektem view vytvoříme novou specifikaci záložky se jménem „account“ a nastavíme mu popisek a přidáme ikonu s názvem „ic_tab_account“. Nakonec specifikaci přidáme do view.

```
Intent intent = new Intent().setClass(this, AccountActivity.class);
TabHost tabHost = getTabHost();
TabHost.TabSpec spec;
spec = tabHost.newTabSpec("account").setIndicator("Account",
res.getDrawable(R.drawable.ic_tab_account))
.setContent(intent);
tabHost.addTab(spec);
```

Seznam účtů

První záložka otvírá activitu s výpisem účtenek. Tato activita dědí od `ListActivity`, což znamená, že v sobě bude obsahovat seznam položek. Ve funkci `onCreate` nainicializuje připojení k SQLite databázi. Následně z databáze získá všechny účtenky, z kterých vytvoří seznam a vloží si ho sám do sebe. Před vložením nad seznamem vytvoří listener, který odchytává možnost mnohonásobného vybrání položek.

Abychom mohli přidávat účtenky, musí si u aktivity vytvořit menu. To vzniká ve funkci `onCreateOptionMenu()`.

```
@Override
public boolean onCreateOptionsMenu(Menu menu) {
MenuInflater inflater = getMenuInflater();
inflater.inflate(R.menu.options_menu, menu);
return true;
}
```

Následně ve funkci `onOptionItemselect()` odchytáváme výběr položky v menu a podle toho vyvoláme akci. Položky tohoto menu se definují pro celou aplikaci. Pokud chceme vytvořit menu pro konkrétní položku v seznamu, musíme vytvořit kontextové menu. Toto menu se aktivuje při dlouhém stisku položky v seznamu. Rovněž musíme řešit odchycení výběru menu a podle toho vyvolat určitou akci.

Aby se vypsal obsah účtenky, musí být zavolaná jiná activita, která tento obsah vypíše. Proto seznam účtenek obsahuje funkci `onListItemClick()`, tak jako parametr dostane celý seznam a pozici prvku na který uživatel klikl. Problémem této funkce je, že se volá i při několikanásobném vybírání položek, proto se v ní musí kontrolovat, jestli daná položka není v tomto problematickém výběru. Pokud není vytvoří se spouštěč s aktivitou pro výpis účtenky, kterému se navíc předá informace o tom, jakou účtenku má vypsat, a activita se spustí.

```
Intent accountContent = new Intent(this, AccountContentActivity.class);
accountContent.putExtra("com.strnava1.mobileWaiter.accountId",
((Account) item).getId());
startActivity(accountContent);
```

V activitě jsou navázané listenery na tlačítkách cancel a ok. Po vybrání stolu a kliknutí na tlačítko OK se spustí funkce, která získá data o stolu a podle toho vytvoří účet. Nejdříve si najdeme ve view prvek podle id, které jme nastavili v šabloně. To nám vrátí objekt typu view proto, abychom z něj mohli získat obsah, musíme ho přetypovat na jeho potomka `Spinner`, což je rozklikávací seznam. Z něho ale získáme typ Object. Jelikož v aplikaci mám již vytvořený objekt `Table`, mohu Object přetypovat a následně z něho získat všechny informace, jako je název a podobně.

```
Table table = ((Table)((Spinner) findViewById(R.id.table))
.getSelectedItem());
```

Upravení účtenky je realizované z kontextového menu. Při odchycení akce „edit“ se otevře nové okno, kde je již předem vyplněný stůl, u kterého je zapsaný upravovaný účet.

Jídelní a nápojový lístek

Obě záložky obsahují stejný kód. V obou případech nejdříve zobrazuji navigaci lístku, tedy kategorie pokrmů nebo nápojů. V tuto chvíli je v menu nastaveno, aby bylo možné přidávat pouze kategorie lístku. V případě kliknutí na položku navigace nevolám novou aktivitu s obsahem určitého druhu, ale pouze změním obsah seznamu na položky v dané kategorii. Nyní akce pro přidání položky vyvolá activitu pro přidání pokrmu nebo nápoje, nikoli přidání kategorie. Zamezil jsem tím vytváření nadměrně rozsáhlých stromů jídelníčku, který by zhoršoval ovladatelnost. Smazání a úprava kategorie nebo jídla a nápoje je realizováno stejným způsobem, jako v případě účtenek. Přepsání seznamu místo otevření nové activity jsem zvolil proto, aby bylo možné rychleji přecházet mezi oběma lístky. Teprve, když se zobrazují položky jídel nebo nápojů, se po kliknutí na položku otevře nová obrazovka s podrobnostmi položky.

Aktivita s podrobnostmi jídla nebo nápoje nabízí uživateli všechny informace, které by mohl host, který si bude objednávat, potřebovat. Zároveň zde uživatel vytvoří objednávku na aktuální položku. V případě, že by více hostů chtělo tuto položku, může uživatel zvýšit množtví položek v objednávce.

SQLite databáze

Databáze je základem celé aplikace. Všechna data, který uživateli aplikace prezentuje jsou v ní uloženy. Hlavní třídou je `DbAdapter`. Obsahuje pouze dvě funkce. První funkce `open()` si v sobě vytvoří instanci třídy `DatabaseHelper`, která fakticky komunikuje s databází a nechá si od ní vrátit přístupnou databázi. Druhou funkcí je funkce `close`, ta pomocí třídy `DatabaseHelper` uzavře databázi. Třída `DatabaseHelper` je potomkem `SQLiteOpenHelper`, získává tím metody volané při vytváření databáze a funkce které mají přístup k databázi. V metodě `onCreate()` dostává parametrem datábázi, v které si nechá třídou `Table` vytvořit tabulky. Metodou `onUpdate`, volané při změně verze databáze, si opět od třídy `Table` nechám odstranit a znovu nahrát databázi. Třída `Table` obsahuje hlavně konstaty s SQL příkazy pro vytvoření tabulek a přidání kořenových položek v navigaci. Jednu položku pro jídelní a druhou pro nápojový lístek.

Ke každé tabulce v databázi existuje jeden adapter. Každý z nich je potomkem `DbAdapter` a je obohacen o funkce pro práci se svojí tabulkou. Pro práci se záznamy jsou v SDK připravené funkce, `insert()` vloží záznam, `update()` aktualizuje řádek odpovídající podmínce ve funkci, `delete()` smaže záznam který odpovídá podmínce a `query()` složí ke klasickému dotazu.

Funkce insert a update vyžadují data uložená v objektu `ContentValues`. Jde o ty obsahující dvojice klíč a hodnota.

```
ContentValues values = new ContentValues();
values.put(KEY_ID, id);
```

Obdobný problém je zpětně při získávání dat z databáze. Funkce query nevrací pole dat, ale vrátí objekt `Cursor`, což je pouze ukazatel na výsledek dotazu. Problémem je, že tento ukazatel ani neukazuje na první položku, proto je nutné před prvním získáváním dat, přesunout ho na začátek. Aplikace, když si požádá adaptér třeba o všechny účtenky, čeká, že adaptér vrátí seznam objektů Account. Bohužel se kvůli objektu `Cursor` není možné obyčejné

přetypování. Proto musí být v každém adaptéru aplikace pro získání dat cursoru a postupné uložení jich do objektů. Tímto způsobem jsem dosáhl jednoduchého ORM.

5.3 Další vývoj systému

Webová aplikace není plně dokončená. Zatím je implementovaná pouze část umožňující správu dat a vzdálenou synchronizaci s mobilním zařízením. Do budoucna je možné systém obohatit o část pro zákazníky. Ti by na stránkách mohli rezervovat místa v restauraci nebo si předem objednat speciální nabídku. Kromě jídelníčku by systém mohl obsahovat správu skladu a automatickou kontrolu zásob. Další implementační částí by mohlo být účetnictví výpisem různých statistik o prosperitě restaurace.

Mobilní aplikace ještě nevyužívá plného potenciálu zařízení, na kterém bude nainstalována. Pro zlepšení práce lze do aplikace přidat upozorňování dlouhém čekání zákazníků. Velkým potencionálem je implementace technologie NFC. Ta by mohla být využita pro identifikaci stolu, u kterého číšník stojí a podle toho vybrat účet zákazníků. Nebo pro využití nově nasazovaného placení bezkontaktní platební kartou.

Další možností je vytvoření v systému chat pro lepší komunikaci zaměstnanců s majitelem podniku. Zaměstnanci by tak mohli snadněji informovat majitele o stavu skladu nebo technických problémech.

Kapitola 6

Testování

6.1 Testování prototypu

Pro testování uživatelského rozhraní využiji Heuristickou evaluaci, pomocí které zjistím nedostatky v návrhu.

Testování proběhlo pouze na jednom expertovi a jednom laikovi. Oba participanti měli splnit dva úkoly.

1. Příhlaste se do systému. Zjistěte, co mají hosté objednané u stolu č. 1 a sdělte to moderátorovi.

2. Dva hosté u stolu č. 6 si chtějí objednat bramborovou polévku. Nyní zadejte objednávku. Poté si host chce objednat Coca-Colu. Vstupte do nápojového lístku až k položce Coca-Cola, ale ještě než ji objednáte, host zruší požadavek. Místo toho si objedná Plzeň 12.

Výsledky testování:

1. Uživatel se nachází v podrobnostech účtenky, zprvu neví, jak přidat objednávku. Čeká tlačítko „Přidat objednávku“ v zobrazených informacích účtenky, na kterou si hosté chtějí objednat.

2. V případě, že se uživatel chce v menu vrátit o jednu úroveň zpět, volí automaticky tlačítko pro přesun na začátek menu, místo aby použil možnost seznamu v horní liště.

Doporučení:

1. Přidat k rozpisu objednávek tlačítko přidat objednávku, která uživatele přesune na jídelní lístek.

2. Zvýraznit možnost vracet se ve stromové struktuře například pomocí tlačítek.

6.2 Testování grafického rozhraní

Testování probíhalo tak, že jsem měl k dispozici dva participanty, kteří měli splnit zadané úkoly. Úkoly jsem vytvořil, abych otestoval funkčnost uživatelského rozhraní. Úkoly v zadání donutili participanta projít co největší částí aplikace. Participant se tedy dostal do stavu, kdy mu v návrhu něco chybělo

Místo testování Testování probíhalo v domácím prostředí. Participanta jsem usadil ke stolu, kde měl připravený mobilní telefon s nahranou aplikací MobileWaiter a úkoly, které měl participant splnit.

Testovací zařízení Aplikace byla nahrána na mobilním zařízení Sony Ericsson Xperia mini pro (Obrázek 6.1) s operačním systémem Android verze 2.3.3. Celé testování jsem zaznamenával na kameru značky JVC.

Obrázek 6.1: Sony Ericsson Xperia mini pro

Průběh testování Participanty jsem nejdříve seznámil s prostředím, ve kterém bude test probíhat a se způsobem testování. Dále jsem je seznámil se zařízením, na kterém test budou plnit, abych snížil jejich případnou nezkušenost s rozhraním systému Android a jeho ovládáním.

Každý z participantů na začátku testu dostal následující seznam úkolů.

Úkoly pro participanty:

1. Zapněte aplikaci MobileWaiter.
2. Přihlašte se do aplikace :
   (a) Zkuste, jestli se tam dostanete bez pomoci.
   (b) Moderátor Vám řekne přihlašovací údaje, zkuste se přihlásit znovu.
3. Podívejte se na podrobnosti stolu 3 a 7.

4. Hosté si chtějí objednat pepsi a gambrinus, šunku s křenem, 2 bramboráčky, kuřecí řízek s hranolkami a vepřové medajlonky s americkými bramborami.

5. Popřejte hostům dobrou chuť a vypněte aplikaci.

Participant 1

Úkol č. 1: Participant po chvíli hledání nalezl aplikaci v seznamu a spustil jí.

Úkol č. 2a: Participant nejdříve hledal nějaké menu, kde by byla položka přihlásit, netušil, že přihlášení právě vidí. Následně se pokusil do uživatelského jména a hesla napsat svoje křesní jméno a přihlásit se. Po zobrazení hlášky, že zadal špatné jméno nebo heslo pokračoval na další úkol.

Úkol č. 2b: Participant požádal moderátora o přihlašovací údaje a zkusil se přihlásit znovu. Nyní úspěšně. Aplikace po přihlášení zobrazila aktivitu se záložkami.

Úkol č. 3: Participant neměl problém se zobrazením podrobností ke stolům.

Úkol č. 4: Participant pochopil zadání tak, že rovnou vstoupil do menu s nápoji, aby začal s objednávkou.Neřešil, k jakému stolu nebo jakým hostům objednávku zapsat. Jelikož participant vlastní mobilní telefon se systémem Android, ovládání dotykem pro něho nebyl problém. Kvůli zkušenostem používal pro návrat zpět v menu hardwarové tlačítko, místo položky v seznamu, čímž ukončil aktivitu se záložkami a vrátil se k přihlašovacímu formuláři. Přihlašovací údaje byly stále zadané, tudíž klikl na tlačítko přihlásit se a pokračoval v plnění úkolu. Tento problém se vyskytl ještě dvakrát, participant se tuto chybu nedokázal odnaučit.

Úkol č. 5: S ukončením aplikace nebyl problém. Tlačítky zpět vypnul aplikaci.

Participant 2

Úkol č. 1: Participant po chvíli hledání nalezl aplikaci v seznamu a spustil jí.

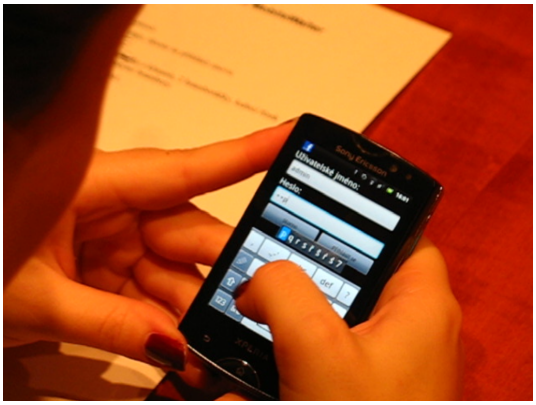
Úkol č. 2a: Participant zkusil náhodné znaky v přihlašovacím formuláři a pokusil se přihlásit. Aplikace ho upozornila na špatné přihlašovací údaje a nepustila ho dál.

Úkol č. 2b: Participant požádal moderátora o přihlašovací údaje a zkusil se přihlásit znovu. (Obrázek:6.2) Nyní úspěšně. Aplikace po přihlášení zobrazila aktivitu se záložkami.

Úkol č. 3: Participant si postupně zobrazil informace o stolu 3 a 7.

Úkol č. 4: Uživatel si přečetl zadání a před zadáváním objednávky chtěl vybrat stůl nebo hosty, ke kterému by objednávku zapsal. Podle instrukcí moderátora pokračoval v zadávání objednávky. (Obrázek 6.3) Po objednání se participantovi objevil Toast s popiskem, že položku objednal, tento popisek byl umístěn u jiného prvku v seznamu než který objednával. U tohoto participanta se znovu objevil problém, kdy v seznamu klikl na hardwarové tlačítko zpět, což vedlo k uzavření aktivity a návratu k přihlašovacímu formuláři.

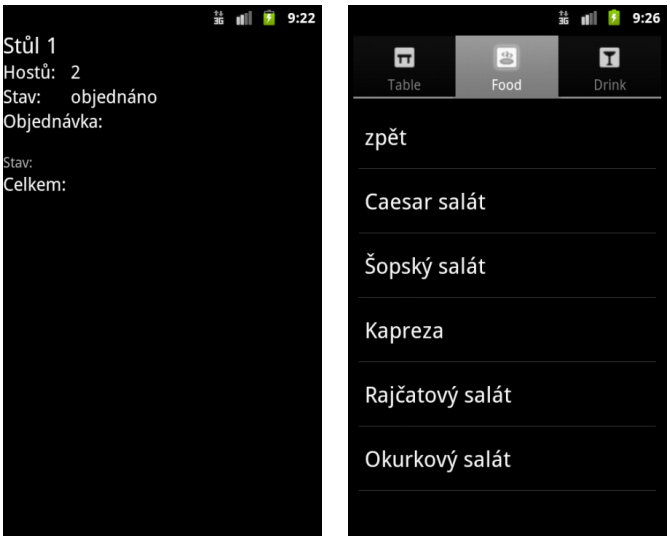
Úkol č. 5: S ukončením aplikace nebyl problém. Tlačítky zpět vypnul aplikaci.



Obrázek 6.2: Druhý pokus o přihlášen

Obrázek 6.3: Pohyb v seznam

Shrnutí testování Testování proběhlo bez větších problémů, jako je např. stabilita programu. V uživatelském rozhraní se našlo několik chyb. Pro následující implementaci aplikace bych doporučil odstranění položky zpět v seznamech (Obrázek 6.4) a převedení její funkce na hardwarové tlačítko, případně na kliknutí na příslušnou záložku. Dále v podrobnostech stolu (Obrázek 6.4) doporučuji přidat tlačítko, které by vedlo k přidání nové objednávky k příslušnému stolu. Zobrazení informace o objednání není dostačující, zpráva, kterou uživatel vidí neobsahuje dostatek informací, které by si spojil s úspěšným objednáním položky. Zároveň je problém v umístění tohoto Toastu (Obrázek 6.5). Možným řešením je změnit formát zprávy z Toastu na nějaký jiný.



Obrázek 6.4: Vlevo: Podrobnosti stolu; Vpravo: Seznam s tlačítkem „zpět“

Obrázek 6.5: Zobrazení Toastu

Kapitola 7

Závěr

Vytvořil jsem základ pro nový systém k usnadnění a zefektivnění provozu restauračních zařízení. Tento systém umožňuje číšníkům založit účet k určenému stolu, provést elektronickou objednávku, odeslat ji do systému a umožňuje zobrazit účtenku.

Pro mobilní aplikaci jsem navrhl prototyp uživatelského rozhraní. Provedl jsem test jeho funkčnosti na dvou participantech, který potvrdil, že i laik dokáže aplikaci snadno ovládat.

Naprogramoval jsem webový server, který spravuje centrální databázi obsahující veškeré informace v systému. Personál restauračního zařízení má tak snadný přístup ke všem informacím o nabízených jídlech a nápojích. Webová aplikace implementuje v sobě vytváření a správu účtů jednotlivých uživatelů, jídelního a nápojového lístku. Systém umí zobrazit objednávky, účtenky. Server umí komunikovat s mobilní aplikací.

Nová mobilní aplikace obsahuje funkcionalitu pro vytváření účtů a objednávek, které se automaticky odesílají do centrální databáze. Plně nahrazuje stále často používané zapisování objednávek na papír a umožňuje rychlý přístup k zadaným informacím dalším uživatelům systému (např. kuchařům), což značně urychlí provoz restaurace. Využil jsem nejnovějších ovládacích prvků Androidu 3.0 a vyššího, aby ovládání aplikace bylo co nejjednodušší.

Funkčnost této aplikace jsem otestoval na třech participantech. Všichni tři dokázali s novou aplikací pracovat úspěšně. Test odhalil několik malých nedostatků. Týkaly se nejasného způsobu přidání objednávky k určenému účtu. Na základě toho zjištění jsem aplikaci rozšířil o tlačítko pro přidání další objednávky. Dále jsem nahradil harwarovým tlačítkem nepříliš vhodně implementovaný způsob návratu ze seznamu jídel do seznamu kategorií.

Výslednou implementaci aplikace se podařilo dostat do stavu, který je uživatelsky přístupný. Aplikace je zprovozněna do té míry, že ji lze nasadit do plného provozu restaurace, přestože se ještě nejedná o finální verzi. Předpokládám, že užití této aplikace v praxi ukáže případné další nedostatky, protože byla otestována pouze v simulovaných podmínkách.

Cíl bakalářské práce byl splněn.

V další fázi vývoje tohoto nového systému se chystám využít co největšího potenciálu mobilního zařízení, jakým jsou vibrace, notifikační lišta, gyroskop nebo technologie NFC. Dalším rozšířením systému je možnost přímého tisknutí účtenek, vytváření statistik, správy skladu, apod.

Zkratky

JRE V angličtině „Java Runtime Environment“.

MySQL Je databázový systém, který pro komunikaci využívá SQL. Systém je veřejný a bezplatný, jelikož není vytvořen pouze pro jeden systém, můžete ho nainstalovat na jakýkoli počítač či server.

NFC V angličtině „Near Field Communication“. Jde o bezdrátovou technologii na krátké vzdálenosti v řádu jednotek centimetrů.

ORM V angličtině „Object-relational mapping“, v češtině známé jako „Objektově relační mapování“. Jedná se o konverzi záznamu v tabulce do odpovídajícího objektu v objektově orientovaném programování.

PC Personal computer

PDA Z angličtiny „Personal Digitam Assistant“, do češtiny přeloženo jako „Osobní digitální pomocník“. Jde o kapesní počítač obvykle ovládaný stylusem, což je malé pero, na který je citlivá dotyková obrazovka.

PHP Jde o rekuzivní z kratku v angličtině „PHP: Hypertext Preprocessor“.

SDK V angličtině „Software development kit“, jedná se o nástroj obsahující předpřipravený kód umožňující programátorům rychlejší vývoj aplikací.

SQL Z angličtiny „Structured Query Language“, jedná se o dotazovací jazyk požívaný v relačních databázích.

Wi-fi Jedná se o bezdrátovou technologii pro komunikaci v počítačové síti.

Obsah přiloženého CD

```
CD
 |-- dokumentace                - Dokumantace bakalářské práce
 |   |--latex                      - Zdrojové soubory textu práce
 |   \--MobileWaiter.eap           - Diagramy projektu
 |   \--bakalarska_prace.pdf       - Text práce v PDF
 |-- prototyp                   - Prototyp vytvořený v Balsamic Mockups
 |-- testovani                  - Vstupní soubory mtx
 |-- zdrojovy_kod               - Zdrojové kódy
 |   |--mobilni_aplkace            - Zdrojový kód mobilní aplikace
 |   |--webova_aplikace            - Zdrojový kód webové aplikace
 \-- MobileWaiter.apk           - Přeložená mobilní aplikace
 \-- readme.txt                 - Informace o obsahu DVD
```

Literatura

[1] VOGEL, L. Tutoriály pro tvorbu aplikací v OS Android.
`http://www.vogella.de/android.html`, stav ze 18. 11. 2011.

[2] web:ab-x. Prodejce restauračních systémů.
`http://www.ab-x.cz`, stav ze 29. 10. 2011.

[3] web:alza. prodejce elektroniky Alza.
`http://www.alza.cz/`, stav ze 8. 5. 2012.

[4] web:balsamiq. Balsamiq Mockups.
`http://balsamiq.com`, stav ze 5. 10. 2011.

[5] web:developer.android. Podpora pro vývoj aplikací v OS Android.
`http://developer.android.com/index.html/`, stav z 5. 1. 2012.

[6] web:nette. Nette Framework.
`http://nette.org`, stav ze 12. 4. 2012.

[7] web:novotny-atrima. Prodejce restauračních systémů.
`http://www.novotny-atrima.com`, stav ze 29. 10. 2011.