Západočeská univerzita v Plzni
Fakulta aplikovaných věd
Katedra informatiky a výpočetní techniky

Diplomová práce

Systém pro správu a realizaci informačních majáku

Plzeň 2013 Bc. Luděk Vlk

Prohlášení

Prohlašuji, že jsem diplomovou práci vypracoval samostatně a výhradně s použitím citovaných pramenů.

V Plzni dne 14. května 2013

Luděk Vlk

Děkuji vedoucímu práce Ing. Ladislavu Pešičkovi za odborné vedení a cenné rady při vypracování diplomové práce.

Abstract

This diploma thesis deals with the problems of providing informations using information beacons. Market of available information systems is explored during this work. Universal system for providing informations to mobile devices was designed and implemented based on discovered problems. In this work was also created a mobile application for iOS and sent for approval to the App Store.

Employees of institutions (zoo, museum, exhibitions, ...) can register their own institution through web interface for free and provide information about their institutions through information beacons. Information beacons is unique identifiable place, that contains informations. Users can find information beacons with mobile app using location, QR codes and 3-digit key.

Obsah

1 Úvod

1.1 Informační majáky

V dnešní moderní době jsou informace dostupné téměř na každém rohu, například na zastávkách, na elektronických panelech nebo i v mobilních telefonech. Největší slabinou je poskytnutí těchto informací lidem, kteří je vyžadují. I v dnešní době, ne každý si umí vyhledat informace, které zrovna potřebuje.

Představme si, co kdyby bylo nepsané pravidlo, že u každého obchodního centra by byla umístěna viditelná nebo neviditelná značka detekovatelná mobilním zařízením (např. telefonem) a poskytla informace o všech slevách nacházejících se v tomto centru. Mohli bychom však i toto pravidlo rozšířit na zastávky hromadné dopravy, kde na základě detekování zastávky mobilním zařízením by se zobrazil seřazený výpis nejbližších spojů od aktuálního času. Pokud by se lidé naučili přijímat takto informace, dostali by jen ty, které je zajímají a pomohou jim v každodenním životě.

Definujeme-li informační maják, jedná se o jednoznačně identifikovatelnou značku, jež poskytne znalostní informace o místě, na němž se nachází. Tyto znalostní informace se ukáží až v případě detekování značky mobilním zařízením. Značka může být viditelná (název, klíč, obrázek) nebo neviditelná (poloha, signál).

1.2 Cíl práce

Cílem diplomové práce je prozkoumat problematiku specializovaných systémů pro poskytování informací návštěvníkům. Na základě zjištěných problému navrhnout a realizovat univerzální systém pro poskytování informací mobilním zařízením. Součástí řešení bude tedy i klientská mobilní aplikace pro platformu iOS (viz 5.1), která bude komunikovat s navrženým systémem, bude schopná detekovat informační majáky a zobrazovat znalostní informace majáků. Klientská mobilní aplikace bude předložena ke schválení digitálnímu obchodu App Store (viz 5.5.1).

2 Systémy pro poskytování informací

Dnes mnoho institucí využívá systémy pro poskytování informací. Obecně se jedná se o levný zdroj informací pro návštěvníky. V této kapitole bude nejdříve popsáno využití těchto systémů v různých odvětvích a následně budou ukázány reálné systémy.

2.1 Význam informačních majáků

Systémy pro poskytování informací můžeme využít v různých odvětví. Můžeme se s nimi setkat na ulici, v obchodech, v muzeích, v zoologické zahradě, ve vzdělávacích institucích nebo i v hromadné dopravě. V této sekci jsou uvedeny příklady, kde se dají systémy pro poskytování informací využít. Systém poskytne informace na základě nalezených majáků mobilním zařízením, respektive pomocí aplikace v mobilním zařízení.

2.1.1 Použití v muzeích

V muzeu se nachází různé exponáty, které jsou umístěny v oddělených sekcích. Muzeum má v digitálním obchodě volně dostupnou aplikaci. Při příchodu do muzea se návštěvníkovi nabídne mapa všech sekcí s jejich krátkým popisem. Každý exponát má čárový kód, který pomocí aplikace návštěvník sejme a obdrží informace v různých jazycích. Různé jazyky cizincům dovolí překonat jazykovou bariéru a mohou si tak lépe užít prohlídku muzea.

Zaměstnancům může tento systém ušetřit práci tím, že nemusejí dělat prohlídky v různých jazycích. Stačí se jen omezit na ty nejrozšířenější jazyky a pro ostatní poskytnout lokalizovaný popis exponátu skrze mobilní aplikaci.

2.1.2 Použití v zoologické zahradě

Zoologická zahrada má několik výběhů s různými exempláři. Při vstupu do zoologické zahrady návštěvník skrze mobilní aplikaci získá mapu všech druhů zvířat a rostlin. Návštěvník navštíví výběh, jenž aplikace rozezná a poskytne

uživateli obrázky a videa o zvířeti. Aplikace zároveň zobrazí uživateli seznam časově nejbližších komentovaných krmení a prezentací zvířat.

Tím, že návštěvník navštíví výběh, zaměstnanci sbírají informace například, který výběh je atraktivní pro návštěvníky, a který naopak je méně zajímavý. Zároveň můžou návštěvníci přidávat komentáře skrze mobilní aplikaci, čímž zaměstnanci zoo získají zpětnou vazbu.

2.1.3 Použití v hromadné dopravě

V úvodu byl zmíněný příklad zastávek hromadné dopravy. Rozšířit ho lze kromě zobrazení časově nejbližších spojů i o vzdálenost potenciálního cestujícího od zastávky a navíc za jak dlouho by mohl u ní normální chůzí být.

2.1.4 Použití ve vzdělávacích zařízení

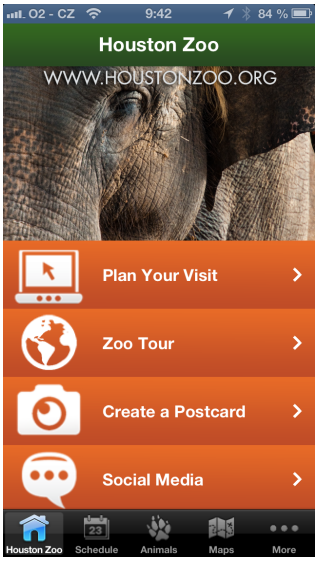
Univerzita se rozprostírá po celém městě, většina fakult má svoji budovu. Informační maják by mohl být umístěn u každé budovy univerzity a nově příchozím studentům by mohl pomoci v orientaci a případně studentovi nalézt jeho výuku, získat jídelní lístek menz či historické údaje o univerzitě.

2.2 Hustonská Zoo

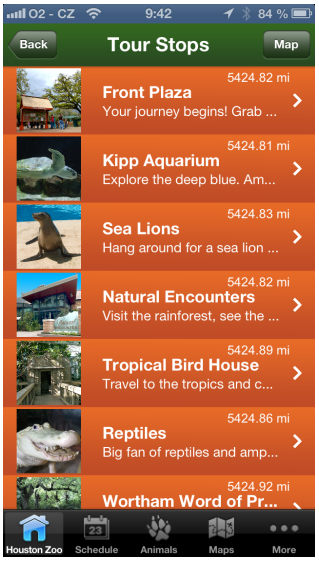
Hustonská Zoo disponuje 6000 zvířat a každý rok ji navštíví 1,84 miliónů návštěvníků. [hus()]

Hustonská Zoo nabízí volně stažitelnou mobilní aplikaci pro systém iOS (viz Obr. 2.1) a Android skrze digitální obchod App Store a Google Play. Aplikace pro obě platformy, jsou co se týče funkcionality, totožné.

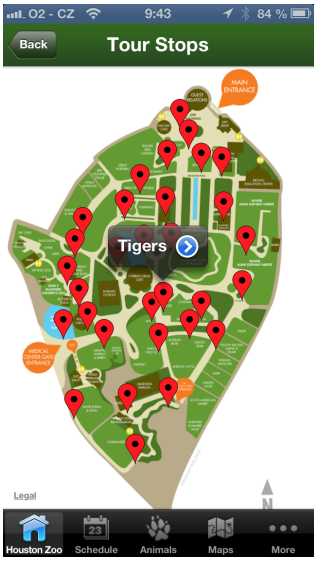
Aplikace nabízí uživateli informace o zoo, její ceník, návštěvní doby a její polohu. Aplikace umožňuje ucelenou prohlídku celé zoo (viz Obr. 2.1b), aniž by musel návštěvník bloudit po zoo a rozhodovat se, jaký výběh navštíví jako další. Při prohlídce nabízí aplikace mapu se všemi možnými výběhy (viz Obr. 2.1c).



(a) Úvodní obrazovka aplikace s rozcestníkem.



(b) Seznam výběhů, kterých se prohlídka týká.



(c) Mapa jednotlivých výběhu, které jsou součástí prohlídky.

Obrázek 2.1: Mobilní aplikace Hustonské Zoo

Aplikace k výběhům nabízí fotografie o exemplářích a krátký popisek. Jako bonus nabízí aplikace vyfocení výběhu s rámečkem.

2.3 Lunchtime

Lunchtime je portál inzerující nejbližší restaurace hladovým zákazníkům, tím že jim ukáže polední menu, které restaurace nabízí.

Princip

Majitelé restaurací si předplatí měsíční inzerci v katalogu Lunchtime. Každý den majitel poskytne portálu denní nabídku jídel své restaurace. Lunchtime tuto nabídku rozšíří skrze svůj portál, email, pdf, RSS, exportuje na web majitele, zobrazí na Facebooku, rozešle dalším portálům a ještě poskytne informace mobilním aplikacím Lunchtime. [lun()]

Pro majitele restaurací se jedná o dobré zviditelnění na internetu,umožní strávníkům zanechat zpětnou vazbu o jejich podniku a automatizuje úkony s denním menu. Lunchtime či hosté se připomenou každý všední den, aby restaurace vyvěsila své polední menu. Tím, že restaurace denně uvolňuje nabídku jídel, se zvyšuje potencionální návštěvnost podniku.

Hostům tento systém umožňuje skrze portál, nebo mobilní aplikaci nabídnout seznam nejbližších podniků s denní nabídkou jídel, na základě které se mohou rozhodnout, jakou restauraci navštíví. Hosté mohou skrze aplikaci provádět rezervace, aniž by museli hledat kontaktní informace na stránkách podniku.

Mobilní aplikace Lunchtime

Aplikace je volně dostupná pro Android, Bada a iOS. Mobilní aplikace umožňuje na základě polohy uživatele zobrazit nejbližší podniky a jejich denní nabídku. Pokud chce uživatel provést rezervaci, lze přistoupit na kartu podniku v aplikaci a získat potřebné informace pro provedení rezervace či zanechání připomínky. Na obrázcích (viz Obr. 2.2) je ukázka z mobilní aplikace pro systém iOS.

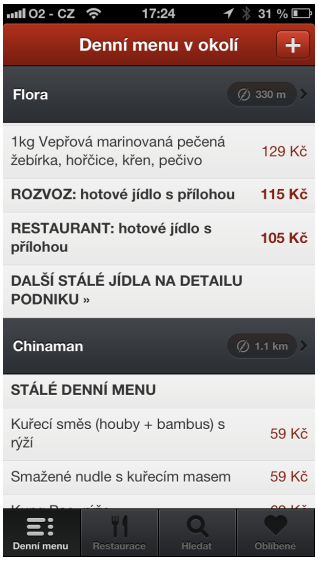
2.4 iZeeum

Průvodce iZeeum je informační systém poskytující informace o umění v muzeích, galeriích a exhibicích skrze mobilní aplikaci pro iOS.

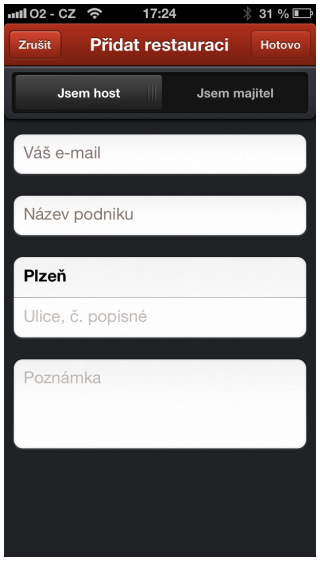
Princip

Zaměstnanci institucí přidají do systému iZeeum informace a fotografie o exponátech a informace o své instituci (popis, poloha, ceník, otevírací doby). Systém iZeeum si fotografie exponátů zpracuje a uloží. Uživatel aplikace vyfotí exponát a fotografie se porovná se systémem iZeeum a podle shody nabídne audiovizuální informace o exponátu. [ize()]

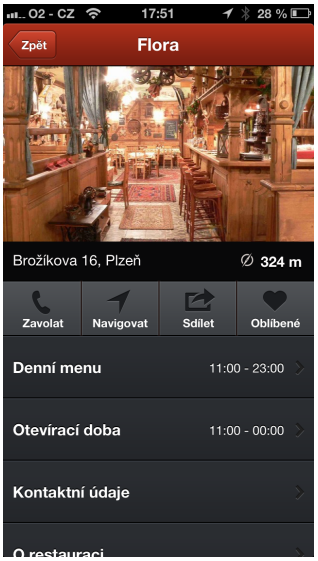
Instituce přidáním se do systému iZeeum získá levný a nenáročný systém pro poskytování informací. Navíc se nenarušuje vizáž exponátů, které by se



(a) Úvodní obrazovka aplikace se seznamem nejbližších podniků.



(b) Skrze mobilní aplikaci lze i přidávat nové restaurace.



(c) Karta podniku.

(d) Mapa nejbližších restaurací.

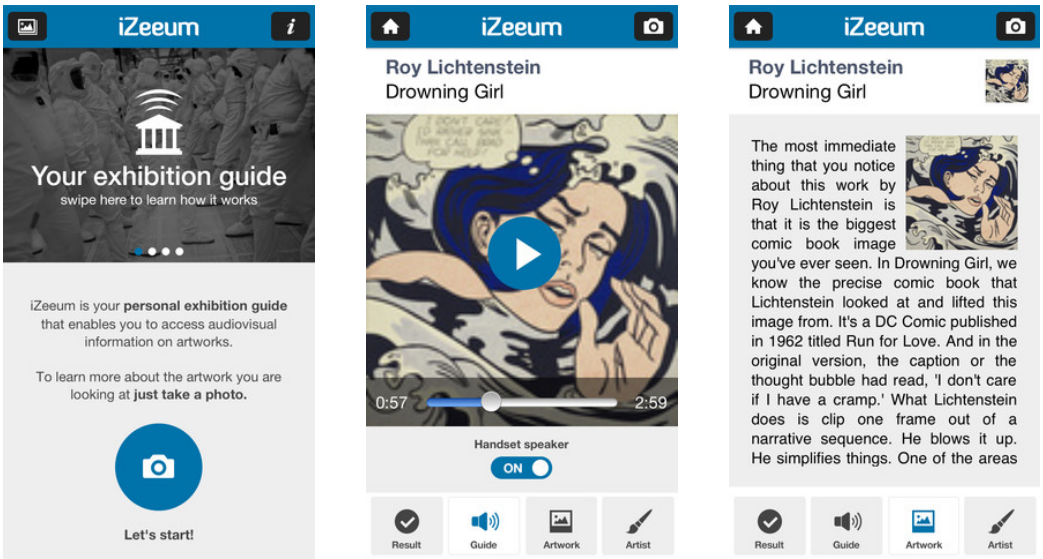
Obrázek 2.2: Mobilní aplikace Lunchtime pro iOS

musely jinak identifikovat pomocí QR kódů, hesel či speciálních obrazů. Tím, že uvnitř muzeí nebo galerií nelze použít lokalizační služby (např. GPS), se jedná o zajímavou alternativu.

Návštěvníkům stačí k identifikaci exponátu jen fotoaparát a připojení k internetu, čímž se zjednodušuje použitelnost této identifikace.

Mobilní aplikace iZeeum

Mobilní aplikace je opět volně dostupná v obchodě App Store (viz Obr. 2.3). Aplikace si zakládá na jednoduchosti a obsahuje jednu obrazovku s informacemi, jak iZeeum funguje a tlačítko pro pořízení fotografie. Na základě vyhodnocené fotografie se zobrazí audiovizuální a textové informace o exponátu. Fotografie se odesílá sytému iZeeum, systém vyhodnotí fotografii a při případné shodě pošle aplikaci informace o exponátu. Systém iZeeum umí vyhodnotit fotografii do 1 vteřiny.



(a) Úvodní obrazovka aplikace s možností pořízení fotografie.

(b) Video o exponátu.

(c) Textové informace o exponátu.

Obrázek 2.3: Mobilní aplikace iZeeum pro iOS

2.5 iTag

Systém iTag poskytuje informace o odjezdech ze zastávek městské hromadné dopravy PMDP v Plzni.

Princip

Na každé zastávce je umístěn NFC tag a QR kód. Pomocí NFC technologie v telefonu nebo čtečky QR kódu se přečte URL[1] adresa, na které je seznam časově nejbližších spojů z dané zastávky. PMDP má tak nový pohodlný způsob poskytnutí jízdního řádu cestujícím. Základním požadavkem je chytrý mobilní telefon disponující mobilním připojením k internetu a fotoaparátem.

[1]Uniform Resource Locator - slouží k přesné specifikaci umístění zdrojů na Internetu.

3 Návrh systému

V této kapitole je popsán návrh univerzálního komplexního systém Tagiee. Pod pojmem instituce se myslí veřejně dostupné zařízení například muzeum, zoo apod.

3.1 Modelový příklad

V této sekci je uvedeno použití systému Tagiee, na základě kterého jsem se rozhodoval při návrhu.

ZOO Plzeň se rozléhá na 21 hektarech, pracuje v ní 134 zaměstnanců a celkem zde můžeme najít 1307 různých zvířat. [zoo()]

V ZOO se budou nacházet desítky informačních majáků. GPS poloha bude využita u venkovních výběhů a v pavilónech QR kód. V dinoparku budou naopak použity NFC tagy. Pomocí těchto technologií lze tyto majáky najít.

Návštěvník navštíví plzeňskou ZOO. Při vstupu bude umístěno značení, že ZOO využívá systém Tagiee a je v něm zaregistrovaná pod unikátním identifikátorem 'zoopl'. Návštěvník si stáhne mobilní aplikaci Tagiee, pokud ji už dávno nemá staženou, z obchodu s aplikacemi. Poté se v aplikaci návštěvník přihlásí k ZOO pomocí seznamu nejbližších institucí nebo pomocí unikátního identifikátoru.

U vstupu se nachází volně dostupná WiFi síť. Uživatel se může přihlásit k této WiFi síti a stáhnout všechen obrázkový obsah, ke kterému se může dostat nalezením majáků a ušetřit tak přenos dat na mobilním připojení.

Návštěvník přijde například k tučňákům, kteří mají svůj venkovní výběh. Návštěvník by se rád dozvěděl další informace o tučňácích, o jejich potravě a stylu života. Pomocí aplikace zapne vyhledávání majáků a aplikace návštěvníkovi nabídne obrázky, videa a informace o tučňácích.

Návštěvník pokračuje v prohlídce a navštíví pavilón plazů. Pavilón plazů je umístěn v budově, ve které není dostupný signál GPS. Vedle terárií jsou však QR kódy. Návštěvník otevře aplikaci a pomocí čtečky QR kódu se mu

zobrazí jednotlivé informace o plazích.

Následující návrh systému by měl být použitelný pro tento modelový příklad (viz 3.3).

3.2 Účel systému

Účelem je tedy navrhnout volně šiřitelný systém pro poskytování informací návštěvníkům institucí. Návštěvník získá na základně navštívených informačních majáků znalosti, ke kterým by se při běžné prohlídce nedostal (např. z nedostatku místa na informačních cedulí). Informace by neměly být návštěvníkům nijak vnucovány, naopak poskytnuty zájemcům pokud možno co nejrychleji a nejintuitivněji.

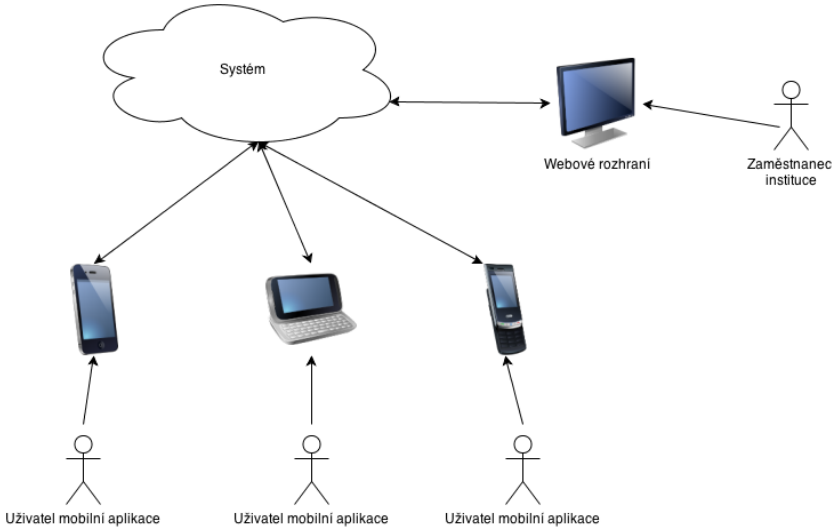
3.3 Funkce a vlastnosti systému

Systém bude uchovávat a poskytovat informace o instituci, o umístění informačních majáků a obsahu majáků při jejich nalezení. Pokud se vrátíme k předešlému motivačnímu příkladu (viz. 3.1), tak instituce je zoo, informační maják bude například zeměpisná poloha výběhu opic a po nalezení tohoto majáku se zobrazí dodatečné informace o opicích.

Systém se bude skládat ze serverové části Tagiee a klientské aplikace Tagiee pro mobilní telefony (viz Obr. 3.1). Systém s mobilní aplikací bude komunikovat pomocí pevně definovaného Web API[1]. Serverová část bude poskytovat webové prostředí, přes které zaměstnanci institucí mohou přidávat a měnit obsah jejich institucí. Klientské aplikace budou dostupné skrze mobilní obchody s aplikacemi (App Store, Google Play, Windows Phone Store). Tagiee může sloužit i jako inovace v oblasti informačních technologií pro instituci.

Zaměstnanec instituce si vytvoří účet v systému Tagiee skrze webové prostředí. K tomuto účtu si přidá jednu nebo více institucí, ke kterým může přidávat různé informační majáky. Informační majáky budou umístěny po

[1]Rozhraní pro komunikaci s webovým serverem pomocí HTTP protokolu a strukturovaných dat.



Obrázek 3.1: Architektura systému Tagiee.

celé instituci a budou detekovatelné pomocí následujících technologií:

- Podle lokace - GPS[2], WiFi[3], Bluetooth[4]. Maják bude nalezitelný nejen podle zeměpisných souřadnic (šířka, výška) ale i podle seznamu nejbližších WiFi a Bluetooth. Poslední dvě technologie usnadní navigaci uvnitř budovy.
- Podle NFC[5] (každý maják může být nalezen pomocí NFC tagu).
- Podle QR kódu[6] (každý maják může být nalezen podle QR kódu).
- Podle 3-místného hash klíče (každý maják může být nalezen podle klíče).

Více o technologiích a metodách nalezení majáků se zabývá Bc. Jiří Hromádka ve své diplomové práci „Komunikace a identifikace informačních majáků“. Po nalezení informačního majáku se uživateli zobrazí obsah v podobě

[2]Global Positioning System - globální družicový polohový systém s jehož pomocí lze určit polohu s přesností na deset metrů.

[3]Označení několika standardů IEEE 802.11 popisující bezdrátovou komunikaci v počítačové síti.

[4]Standard pro bezdrátovou komunikaci mezi dvěma či více elektronickými zařízeními.

[5]Near field communication - slouží ke komunikaci mezi dvěma zařízeními na krátkou vzdálenost.

[6]Prostředek pro automatizovaný sběr dat.

textů, obrázků a videa. Zaměstnanci budou mít přístup ke statistikám o návštěvnosti jejich instituce a zpětnou vazbu od uživatelů pomocí krátkých hodnocení, která mohou návštěvníci přidat k instituci.

Uživatel si bude moci zdarma stáhnout klientskou aplikaci do svého telefonu. Při zapnutí aplikace se uživateli na základě jeho polohy nabídne seznam nejbližších institucí, ke kterým se může přihlásit. Přihlášení k instituci lze i zadáním unikátního identifikátoru, jenž má každá instituce v systému Tagiee. Po přihlášení k instituci může uživatel zobrazit důležité informace jako jsou otevírací doby, ceník, kontaktní údaje. Uživatel pomocí aplikace může naleznout informační majáky pouze u přihlášené instituce. Nalezením majáku získá znalostí informace o místě, kde se maják nachází. Nalezené majáky se ukládají pro pozdější prohlížení.

Mobilní zařízení jako poskytovatelé těchto informací byla vybrána z důvodu jejich rozšířenosti mezi návštěvníky. Dle studie ve 3. čtvrtletí roku 2012 bylo prodáno více než 1083 miliónů „chytrých“ mobilních zařízení. [stu()] Mobilní „chytré“ zařízení využívá fotoaparát, lokalizační služby, mobilní internet, umožňuje instalaci aplikací třetích stran apod.

Systém a klientská aplikace budou mít tedy následující vlastnosti a funkčnost (viz 3.3.1 a 3.3.2).

3.3.1 Systém

- Systém bude navržen pro snadné budoucí rozšíření.
- Systém bude disponovat webovým rozhraním pro zaměstnance institucí.
- Systém bude komunikovat s klientskou aplikací skrze Web API.
- Umožní zdarma registraci do systému Tagiee.
- Umožní přidání a editování institucí, majáků přihlášeným uživatelům.
- Obsah jednotlivých majáků budou tvořit texty, obrázky a video, které bude umět systém přidat a editovat.
- Systém bude uchovávat veškeré textové informace a obrázky. Videa budou uložena na externích úložištích.

- Systém poskytne seznam nejbližších institucí na základě polohy uživatele.
- Systém poskytne klientským aplikací všechen obsah přidaný zaměstnanci institucí.
- Systém musí jednoznačně identifikovat všechny zařízení, na kterých běží klientská aplikace, instituce, majáky a zdroje (obrázky, videa apod.).
- Systém bude přijímat informace z klientské aplikace a na základě nich tvořit statistiky o využití Tagiee v institucích.
- Systém bude ukládat informace o nalezení informačního majáku. Pokud uživatel smaže klientskou aplikaci, tak po znovu nainstalování aplikace nepřijde o jednou již nalezené majáky.
- Systém bude přijímat hodnocení uživatelů, která budou sloužit jako zpětná vazba zaměstnancům instituce.

3.3.2 Klientská aplikace

Klientská aplikace by pokud možno měla mít následující funkčnost, aby se zachovala úroveň služby Tagiee.

- Mobilní aplikace bude splňovat podmínky pro vstup do digitálních obchodů s aplikacemi (App Store, Google Play, Windows Phone Store).
- Aplikace bude využívat GPS, NFC, WiFi, Bluetooth, fotoaparát a připojení k internetu.
- Klientská aplikace bude komunikovat se systémem Tagiee pomocí API.
- Aplikace bude schopná zobrazit nejbližší instituce využívající systém Tagiee a následné přihlášení k nim. Pokud nebude dostupná GPS technologie pro získání aktuální polohy, bude možnost se přihlásit k instituci pomocí unikátního identifikátoru. Přihlášením se míní stažení informací o instituci včetně všech dostupných majáků a jejich textového obsahu. Stažené textové informace se uchovají pro pozdější použití.
- Umožní nalezení informačních majáků návštěvníkem pomocí jednoznačné identifikovatelné značky (lokalizační služba - GPS, QR kód, NFC, WiFi, bluetooth, 3-místný hash klíč).

- Při nalezení majáků skrze lokalizační službu, musí aplikace zobrazit všechny nalezitelné majáky na mapě.
- Aplikace bude umět zobrazit seznam všech nalezených majáků, jejich obsah včetně všech obrázků a videí. Jednou již stažené obrázky si aplikace uchová, aby se v budoucnu nestahovaly znovu.
- Mobilní aplikace Tagiee musí být šetrná k využití mobilní datové sítě. V případě připojení k WiFi síti je možnost před začátkem prohlídky instituce stáhnout všechny obrázky obsahu majáků, které se zobrazí v případě jejich nalezení.
- Aplikace musí zobrazit vždy aktuální informace, pokud je dostupné připojení k internetu.
- Klientská aplikace bude sbírat statistiky o akcích uživatele. Akce jsou přihlášení k instituci a nalezení majáku. Tyto akce bude umět poslat systému Tagiee. Pokud nebude dostupné připojení k internetu, tyto statistiky se uloží a pošlou, až připojení k internetu bude znovu navázáno.
- Aplikace bude umět přidat krátké hodnocení.
- Velikost celé aplikace musí být minimalizována pro možnost stažení i přes mobilní internetové připojení.
- Aplikace musí být šetrná k baterii a zapínat / vypínat lokalizační služby dle potřeby.
- Aplikace musí unikátně napříč platformami identifikovat zařízení.

3.4 Uživatelé systému

K systému budou přistupovat 3 typy uživatelů. Zaměstnanci institucí, návštěvníci využívající klientskou aplikaci a samotní správci systému Tagiee.

3.4.1 Zaměstnanec instituce

Zaměstnanec instituce bude pověřená osoba, která bude tvořit obsah své instituce v systému Tagiee. Zaměstnanec bude schopen vykonávat tyto úkony.

- Bezplatná registrace do systému Tagiee.
- Přidání / editování institucí.
- Přidání / editování majáků a jejich obsahu. S tím spojené nahrávání obrázků a videí.
- Prohlížení statistik návštěvnosti a hodnocení od uživatelů.
- Nahlášení nevhodného hodnocení od uživatelů.

3.4.2 Uživatel klientské aplikace

Uživatel musí vlastnit mobilní „chytrý“ telefon, který disponuje připojením k internetu.Uživatel bude schopen vykonávat tyto následující úkony.

- Bezplatně stáhnout klientskou aplikaci Tagiee z aplikačních obchodů.
- Uživatel může povolit užívání lokalizační služby v aplikaci.
- Nalezení informačních majáků pomocí lokalizačních služeb, QR kódu, 3-místného hash klíče a NFC tagu.
- Přidání krátké recenze k instituci.
- Stažení všech obrázků před prohlídkou instituce.
- Přihlášení k instituci skrze její unikátní identifikátor nebo na základě polohy.
- Zobrazit informace o instituci.
- Zobrazit již jednou nalezené majáky s jejich obsahem.

3.4.3 Správce Tagiee

Správce systému Tagiee se musí starat o kvalitu obsahu přidaných zaměstnanci institucí.

- Kontrolovat pravost institucí, aby nedocházelo k znekvalitnění služby Tagiee.

- Mazat nevhodná (vulgární a nemravná) hodnocení uživatelů na základě nahlášení zaměstnanci instituce.

3.5 Návrh možných alternativ

Alternativou může být odstranění přihlášení k instituci. Mobilní aplikace by po zapnutí začala automaticky vyhledávat nejbližší majáky, aniž by byla definována instituce. Zjednodušila by se klientská aplikace, jež by po startu odeslala informace o své poloze systému Tagiee, který by poskytl informace o nejbližších majících. Problémem zde může být míchání jednotlivých majáků různých institucí a jejich ověření pravosti. Není problém pak nastrčit maják do instituce, který má mylné informace.

Současný navržený systém tohle odstraňuje tím, že si uživatel sám vybere, k jaké instituci se chce přihlásit. Aplikace totiž rozpozná pouze ty majáky, které přidali zaměstnanci instituce.

4 Realizace systému

Pro realizaci systému byla vybrána technologie MySQL (viz 4.2.2) a PHP (viz 4.2.1) pro rychlý vývoj systému. API pro komunikaci s klientskými aplikacemi bylo napsáno v čistém PHP. Pro vývoj frontendového webového prostředí byl použit český Framework Nette (viz 4.2.3).

4.1 Umístění systému

Systém Tagiee byl umístěn na placeném webhostingu Onebit pod doménou tagiee.com. Hosting podporuje PHP 5.3/5.4, MySQL.

4.2 Použité technologie a knihovny

4.2.1 PHP

PHP je technologie běžící na serveru. Stránky se píší pomocí php skriptů, které obsahují jak HTML, CSS a kód PHP. Ve chvíli, kdy webový server obdrží požadavek na zobrazení stránky využívající PHP skripty, server vezme HTML kód, poté server provede části PHP skriptů a výsledek spojí do odpovědi. [php()]

4.2.2 MySQL

MySQL je relační databázový systém typu SŘBD[1]. Každá databáze v MySQL je tvořena z jedné nebo více tabulek, které mají řádky a sloupce. V řádcích jsou jednotlivé záznamy a sloupce uvozují název či datový typ jednotlivých polí. Práce s daty se provádí pomocí deklarativního programovacího jazyka SQL.

[1]Systém řízení báze dat - software zajišťující práci s databází.

4.2.3 Framework Nette

Framework Nette je opensource framework pro tvorbu webových aplikací v PHP 5. Výhody frameworku jsou eliminace bezpečnostních rizik, šablonovací systém, podpora AJAXu, čistý objektový návrh, komponenty, událostmi řízené programování, výkon, dependency injection, pluginy a rozšíření.

Nette využívá architekturu MVP:

- Model - Datová a funkční vrstva, jež se stará o ukládání dat a aplikační logiku (např. přihlášení, přidání a změna dat).
- View - Stará se pouze o vykreslení výsledku požadavku uživatele. V Nette jsou to šablony.
- Presenter - Spojuje Model a View vrstvu. Na základě požadavku od uživatele vyvolá příslušnou aplikační logiku.

Informace o Nette v této podsekci vychází z její dokumentace. [net()]

Šablony

Silnou předností Nette je šablonovací systém Latte. Latte ušetří práci a zabezpečí výstup například proti XSS[2] apod.

Základem je Latte soubor, který obsahuje HTML, Javascript a Latte makra. V následujícím příkladu je uveden rozdíl mezi PHP a Latte makry. V PHP bude vypsáno pole prvků jako seznam, kde každý řetězec bude začínat velikým písmenem:

```
<?php if ($items): ?>
    <?php $counter = 1 ?>
    <ul>
    <?php foreach ($items as $item): ?>
        // Tvoreni ID itemu.
        <li id="item-<?php echo $counter++ ?>">
        <?php
        // Prvni znak retezce velke pismeno.
```

[2]Cross-Site Scripting je metoda narušení webových stránek zneužívající neošetřených výstupů. Útočník pak dokáže do stránky podstrčit svůj vlastní kód a tím může stránku pozměnit nebo dokonce získat citlivé údaje o návštěvnících.

```
        echo htmlSpecialChars(
                        mb_convert_case($item, MB_CASE_TITLE))
        ?>
        </li>
    <?php endforeach ?>
    </ul>
<?php endif?>
```

Tento kód je však velmi nepřehledný. Pomocí Latte maker bude kód vypadat následovně:

```
<ul n:if="$items">
    <li n:foreach="$items as $item">{$item|capitalize}</li>
</ul>
```

Všimnout si můžete tzv. helperů, jsou to slova za znakem „|“, které umožňují formátování výsledného textu, čísel apod. V předchozím příkladě je helper slovíčko capitalize, které znamená, že řetězec bude vypsán vždy s prvním velkým písmenem.

Presentery

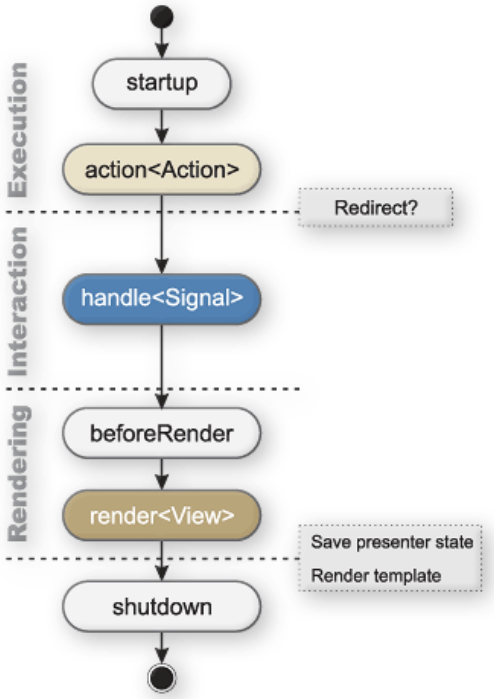
Presenter je objekt, jenž vezme požadavek přeložený routerem (obousměrné překládání mezi URL a akcí presenteru) a vybere odpověď. Odpovědí může být HTML stránka, obrázek, XML dokument, soubor na disku, JSON, přesměrování apod. Každý presenter má svůj životní cyklus (viz obr. 4.1).

Ihned po vytvoření presenteru se zavolá metoda startup(). Ta inicializuje proměnné nebo oveří uživatelská oprávnění. Metoda action provede určitý úkon (přihlásí uživatele, zapíše data do databáze) případně přesměruje na jiný presenter. Metoda handle zpracovává signály. Signály jsou určeny pro komponenty a pro zpracování AJAXových požadavků. Důležitou metodou je render, při níž předá presenter šabloně data, která bude šablona zobrazovat.

Databáze

Nette poskytuje vrstvu pro práci s databází, jež za programátora bude pokládat efektivní dotazy, které nebudou přenášet zbytečná data. Základem je objekt nad PDO[3] *Nette/Database/Connection*. Nad tímto objektem se dají

[3]Třída reprezentující spojení mezi PHP a databázovým serverem.



Obrázek 4.1: Životní cyklus presenteru v Nette.

pokládat klasické SQL dotazy Na následujícím příkladu je ukázáno vytvoření objektu Connection a a položení SQL dotazu.

```
use Nette\Database\Connection;

$database = new Connection($dsn, $user, $password);
$database->query('SELECT * FROM categories WHERE id=?', 123)
                ->dump();
```

U složitějších dotazů lze využít nástroj odvozený od NotORM. NotORM je PHP knihovna pro snadnější práci s databází a obsahuje balík metod, které za nás vygenerují požadované SQL dotazy. Nutné je však podotknout, pokud bychom chtěli využívat tuto funkci v Nette, je nutné mít typ tabulek v databázi s cizími klíči např. InnoDB, kde název sloupečku cizích klíčů bude obsahovat název tabulky, na kterou se odkazuje a suffix id. Na následujícím příkladu je použití nástroje NotORM.

```
use Nette\Database\Connection;

$database = new Connection($dsn, $user, $password);
$database->table('categories')->where(array('id'=>123))
                ->dump();
```

Tímto způsobem můžeme tvořit mnohem složitější dotazy a využívat různé agregační a filtrační funkce. Požadovaná data jsou vrácena jako ActiveRow objekt, se kterým zacházíme jako s polem.

Formuláře

Nette umí jednoduše vytvořit formuláře třídy *Nette/Forms/Form*. Formuláře se validují jak na straně serveru, tak i na straně Javascriptu, podporují vícejazyčnost a poskytují zabezpečení proti zranitelnosti systémů. Jednoduchý přihlašovací formulář vypadá následovně.

```
require 'Nette/loader.php';

use Nette\Forms\Form;

$form = new Form;
// Pridani poli
$form->addText('name', 'Jmeno:');
$form->addPassword('password', 'Heslo:');
$form->addSubmit('send', 'Registrovat');

echo $form; // vykresli formular
```

Nette má velikou škálu validačních pravidel od emailu až po URL. V případě úspěšně vyplněného formuláře se zavolá metoda, která se formuláři nastaví následovně.

```
$form->onSuccess[] = $this->loginSubmited;

public function loginSubmitted(Form $form) {
        // Zpracovani
        $jmeno = $form->values->name; // Jmeno
        $password = $form->values->password; // Heslo
}
```

Metoda *loginSubmitted* bude zavolána při odeslání formuláře. K jednotlivým hodnotám z formuláře se přistupuje přes instanci Form předanou jako parametr této metody.

4.2.4 Google Maps JavaScript API v3

Google Maps JavaScript API v3 umožňuje vložení Google mapy na vlastní stránky. API poskytuje funkce které pracují s mapou a mohou jí nastavovat různé anotace, polohu, ale i informace z mapy získávat. [map()]

Mapa se připojí k blokovému elementu <*div*> pomocí javascriptového kódu.

```
var map;
function initialize() {
  var mapOptions = {
    zoom: 8,
    // Nastaveni jake misto na mape ma byt zobrazeno.
    center: new google.maps.LatLng(-34.397, 150.644),
    // Typ mapy
    mapTypeId: google.maps.MapTypeId.ROADMAP
  };
  // Napojeni mapy na div s id="map-canvas"
  map = new google.maps.Map(document.getElementById('map-canvas'),
      mapOptions);
}

// Inicialize Google Map
google.maps.event.addDomListener(window, 'load', initialize);
```

4.2.5 Google Charts

Google Charts je nástroj, který umožní vložení grafů na stránky pomocí javascriptu. Podporuje veliké množství grafů např. koláčové, spojnicové, sloupcové apod. Nejprve je nutné inicializovat graf. [cha()]

```
// Nacteni vizualizace grafu.
google.load('visualization', '1.0', {'packages':['corechart']});

// Nastaveni callbacku pro vykresleni grafu.
google.setOnLoadCallback(drawChart);
```

Po inicializaci se nadefinují data a naplní se jim instance grafu.

```
function drawChart() {
       // Definice dat.
       var data = new google.visualization.DataTable();
       data.addColumn('string', 'Topping');
       data.addColumn('number', 'Slices');
```

```
    data.addRows([
      ['Mushrooms', 3],
      ['Onions', 1],
    ]);

    // Nastaveni grafu.
    var options = {'width':400,
                   'height':300};

    // Vytvoreni instance grafu, ktere se predaji
    // data a nastaveni.
    var chart = new google.visualization.PieChart(
            document.getElementById('chart_div'));
            // Vykresleni grafu.
    chart.draw(data, options);
  }
```

4.2.6 JQuery

JQuery je javascriptová knihovna usnadňující práci a přehlednost javascriptu. [jqu()] Knihovna má následující funkce:

- Výběr DOM elementů.
- Funkce pro procházení a změnu DOM.
- Kompletní podpora AJAX.
- Manipulace s CSS.
- Události, na které lze napojit vlastní funkce.
- Rozšiřitelnost o další pluginy.

Použití knihovny spočívá v importování javascriptové knihovny do stránek:

```
<script type="text/javascript" src="jquery.js"></script>
```

4.2.7 JQuery UI

JQuery UI je sada efektů, widgetů, témat a interaktivních prvků postavená nad knihovnou JQuery. Základem je importování CSS a javascriptového souboru do HTML dokumentu a poté lze využívat funkce tohoto rozšíření. [jqu()]

4.2.8 Uploadify

Uploadify je JQuery plugin, pomocí kterého lze nahrávat na server soubory. Jsou dvě různé verze a to HTML5 a Flash. Uploadify podporuje nahrání více souborů, zobrazuje průběh nahrávání souboru a v HTML5 verzi podporuje drag and drop přidání souboru do nahrávacího formuláře. Zobrazení formuláře pro nahrávání souborů se provede nasazením pluginu pomocí JQuery nad blokovým elementem <*div*>. [upl()]

```
$(function() {
    $("#file_upload").uploadify({
        width         : 120,
        height        : 30,
        // Nahravaci animace
        swf           : '/uploadify/uploadify.swf',
        // PHP skript pro zpracovani nahraneho obrazku.
        uploader      : '/uploadify/uploadify.php'
    });
});
```

4.2.9 Jcrop

JCrop je JQuery plugin, který umožní vybrat výseč v obrázku. Nad elementem <*image*> nebo <*img*> se nasadí JCrop plugin. [jcr()]

```
$(function() {
        $('#jcrop_target').Jcrop();
});
```

4.3 Databázová vrstva

Databáze v MySQL byla vytvořena pomocí nástroje MySQL Workbench 5.2. MySQL Workbench je grafický uživatelský nástroj pro architekty a vývojáře databází. Umožňuje tvorbu a modelování databází MySQL. Na obrázku (Obr. 4.2) je datový model systému Tagiee.

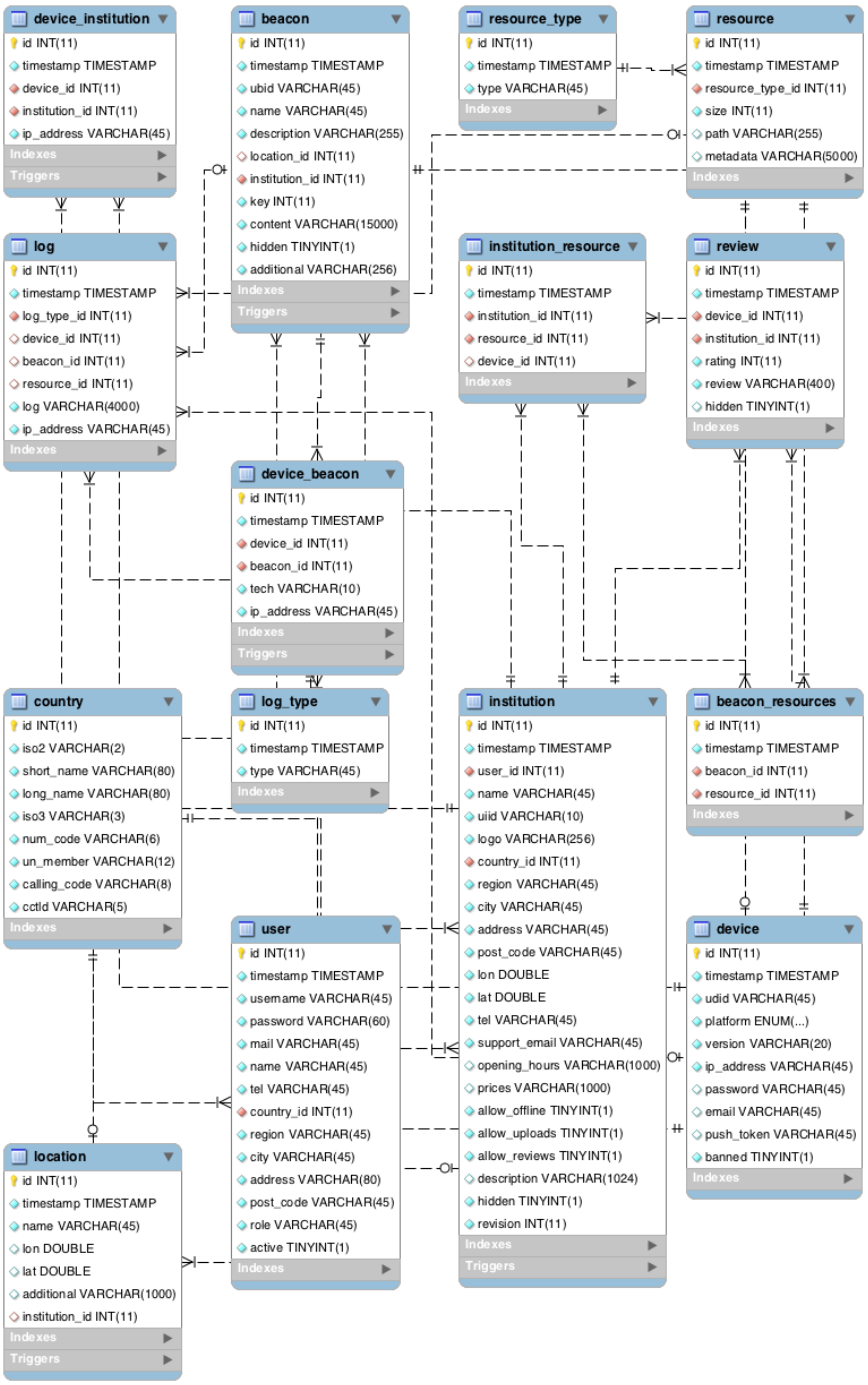
Názvy všech tabulek jsou anglicky malým písmenem a v jednotném čísle. Propojovací tabulky cizích klíčů jsou tvořeny názvy propojených tabulek oddělené podtržítkem. Aby se databáze mohla využívat s frameworkem Nette, všechny cizí klíče jsou pojmenovány názvem tabulky, na kterou se odkazují se suffixem id.

Základem je tabulka uživatelů (user). Zaměstnanec instituce si musí nejprve vytvořit tento účet, kde zadá informace o sobě. Uživatel má různé role, pokud se jedná o správce instituce, dovolí systém Tagiee přidání jedné či více institucí (institution). K přidání instituce stačí název instituce, unikátní identifikátor instituce (uiid) a adresa. Adresa se skládá ze země, kraje, města, ulice a poštovního směrovacího čísla. Na základě těchto údajů se získají zeměpisné souřadnice z Google Maps. Nově vytvořená instituce není aktivní a je nutné ji aktivovat pro zobrazení v mobilních aplikacích. Aktivovat instituci lze jen v tom případě, že jsou vyplněny položky: popis instituce, ceník, otevírací doby a je přidán alespoň jeden maják. Ceník a otevírací doby jsou uloženy v databázi jako JSON (viz 4.4.1) pole.

K instituci lze přidat majáky (beacon), zdroje (resource) a lokace (location). Zdroj je multimediální obsah, jako je například obrázek. Zdroje jsou propojeny s institucí pomocí propojovací tabulky resource_institution. Lokace je umístění informačního majáku. Nemusí se jednat pouze o zeměpisné souřadnice, ale i o seznam nejbližších WiFi a Bluetooth sítí, podle kterých lze jednoznačně definovat polohu informačního majáku. Seznam WiFi a Bluetooth je v databázi uložen jako JSON pole.

Instituce může mít libovolné množství majáků. Maják má unikátní identifikátor ubid a 3-místný klíč, který je unikátní jen v rámci instituce. QR kód a NFC budou využívat ubid k identifikaci majáku a klíč bude využívat 3-místného klíče. Každý maják má svůj popis a textový obsah. Obsah je uložen opět jako JSON pole v následujícím formátu:

```
[1..N array] =>
        (
                [type] => String {image|text|audio|video}
```



Obrázek 4.2: Datový model systému Tagiee.

```
[heading] => String
```

```
            [content] => String
            [resource_id] => Array integer
    )
```

Rozlišují se 4 typy obsahu a to obrázek, video, odkaz a text. V případě textu se vyplní kolonky heading, content a type se nastaví na text. U odkazu se vyplní také heading a content, kde v heading bude název odkazu a v content bude přímo odkaz. U obrázků se může heading a content vynechat, ale musí být vyplněn resource_id. V resource_id bude pole id referující na obrázky. U videa bude pouze v content odkaz na Youtube video. Jeden maják může mít tedy „neomezené“ množství textů a odkazů (omezení dáno velikostí sloupečku v databázi), 10 obrázků a 1 Youtube video. O tohle omezení se stará webová aplikační vrstva systému Tagiee. Zdroje k majáku jsou zaznamenány v tabulce beacon_resource.

Každé mobilní zařízení se zařadí do tabulky device s unikátním identifikátorem. Přihlásí-li se zařízení k instituci, vloží se záznam do tabulky device_institution a v případě nalezení majáku se vloží záznam do beacon_device.

Tabulka review uchovává hodnocení uživatelů k dané instituci. Jedno zařízení může přidat pouze jedno hodnocení k jedné instituci.

4.3.1 Triggery

Od verze MySQL 5.0.2 přibyla podpora trigerů. Triger je pojmenovaný databázový objekt, který je spojen s určitou tabulkou a aktivuje se událostí při práci nad ní. MySQL podporuje události INSERT (vložení), UPDATE (aktualizace dat) a DELETE (smazání). Ještě se rozlišují akce BEFORE (před vykonání události) a AFTER (po vykonání události). [mys()]

Při vykonání triggeru mohou být inicializovány dvě proměnné a to NEW a OLD. Při události INSERT může být použita pouze proměnná NEW. OLD proměnná v události INSERT není. U události DELETE je možno použít zase naopak pouze proměnnou OLD. Nakonec u události UPDATE lze použít obě proměnné. Proměnná OLD je pouze určená ke čtení a nelze do ní zapisovat. [mys()]

Revize instituce

Každá instituce má číslo svojí revize. Ta určuje aktuálnost dat pro klientské aplikace. Pokud má klientská aplikace číslo revize nižší než je v systému, aplikace ví, že má zastaralá data a stáhne si poslední verzi dat. Revize se bude zvyšovat o číslo 1 při každé změně / přidání informací v instituci a jejich majících. Tuto logiku jsem přesunul z aplikační vrstvy do databázové. V databázi je nad tabulkou institution, beacon a location trigger, který zajistí zvyšování čísla revize v instituci. Na následujícím příkladě je trigger, který po každé změně informací nad tabulkou beacon inkrementuje revizi instituce.

```
DELIMITER //
CREATE TRIGGER ‘update_rev_beacon‘ AFTER UPDATE ON ‘beacon‘
 FOR EACH ROW BEGIN
                UPDATE institution i
                SET i.revision = i.revision + 1
                WHERE i.id = NEW.institution_id ;
        END //
DELIMITER ;
```

Trigger se spustí při aktualizaci každého záznamu v tabulce beacon díky klíčovému slovu FOR EACH ROW. Podobné triggery jsou v tabulkách location a institution. Při přidání hodnocení uživatelem, tedy aktualizace tabulky review, nedochází ke změně revize. Hodnocení nejsou klíčové informace, pro která by bylo nutné stahovat všechny informace o instituci znovu.

Návštěvnost majáků, institucí

O návštěvníkovi, jenž navštíví instituci, se do databáze tato událost zaznamená. Návštěvník může navštívit instituci vícekrát. Aby se neplnily statistiky, tak navštívení instituce bude zaznamenáno jen jednou za 24 hodin. Pokud by návštěvník navštívil instituci ve 24 hodinách vícekrát, v databázi bude záznam pouze jednou. Naopak navštíví-li instituci dvakrát s intervalem jednoho měsíce, budou zaznamenány v databázi dvě návštěvy. V následujícím kódu je ukázka vytvoření triggeru nad tabulkou device_institution, který nepovolí přidání více záznamů o navštívení stejné instituce stejným návštěvníkem během 24 hodin.

```
DELIMITER //
CREATE TRIGGER ‘device_institution_prevent‘ BEFORE INSERT ON
 ‘device_institution‘ FOR EACH ROW BEGIN
```

```
DECLARE count INT default 0;

        SET count =
                (SELECT COUNT(*)
                FROM device_institution
                WHERE
                        device_id = NEW.device_id
                AND
                        institution_id = NEW.institution_id
                AND
                        abs(UNIX_TIMESTAMP(NEW.timestamp)
                        -
                        UNIX_TIMESTAMP(timestamp)) < 86400 );


        IF(count > 0) THEN
                SIGNAL SQLSTATE '45000'SET
                MESSAGE_TEXT = "device already visit
                institution in this day!";
        END IF;
END
//
DELIMITER ;
```

Před vložením nového záznamu se projde tabulka device_institution a porovná se, kolik záznamů v tabulce je přidaných do 24 hodin (86400 sekund) od právě přidávaného záznamu v proměnné NEW. Pokud takový záznam existuje, jedná se o duplicitu za 24 hodin a je vyvolána výjimka, která zruší přidání záznamu.

Stejným způsobem je řešeno nalezení majáku návštěvníkem. Objevení je zaznamenáno pouze jednou za 24 hodin, stejně jako u navštívení instituce. Kód triggeru je obdobný a nachází se nad tabulkou device_beacon.

4.4 Tagiee API

Systém především komunikuje s mobilními zařízeními. Pro implementaci bylo použité čisté PHP. Komunikace probíhá formou dotaz / odpověď, tedy na každý dotaz je odpověď. Komunikace je skrze HTTP Post požadavek a samotná data jsou v JSON formátu (viz 4.4.1).

4.4.1 JSON format

JSON je formát dat určený především k jejich přenosu. Je jednoduchý na generování a parsování. V JSON se vyskytují dvě struktury:

- Kolekce neseřazených párů (název a hodnota) nazýváno jako objekt. Objekt začíná závorkou { a končí párovou závorkou }. Za každým názvem je dvojtečka a páry se oddělují čárkou.
- Seřazené pole prvků. Pole začíná hranatou závorkou [a končí druhou párovou závorkou]. Prvky v poli jsou odděleny čárkou.

Struktury se můžou do sebe libovolně vnořovat a tvořit tak spolu komplexnější datovou strukturu. Hodnotou může být řetězec, číslo, pole, objekt, true, false a null. [jso()]

4.4.2 API pro komunikaci

Jak už bylo zmíněno, požadavek je v JSON formátu. Server získá požadavek pomocí funkce *file_get_contents()* nad *php://input*.

```
$input = file_get_contents('php://input');
```

Funkce *file_get_contents($file)* přečte celý soubor nad vstupním streamem. V případě POST požadavku jsou vstupním streamem data, která POST přenáší, tedy v našem případě JSON struktura. Otevřený stream smí být čten pouze jednou a nepodporuje operace pro vyhledávání.

Tímto jsme dostali JSON strukturu, jež je nutné dekódovat. Dekódování je provedeno nativní funkcí PHP json_decode($json).

```
$in = json_decode($input);
```

Pokud není nastaveno, výsledek po parsování je v objektu, kterému rozumí PHP a můžeme s ním dále pracovat.

Každý požadavek musí obsahovat pole *type*, které určuje, jaký obsah má být vrácen. Další povinný parametr je *udid*, jenž je unikátní identifikátor každého zařízení. Spolu s unikátním identifikátorem je povinným parametrem

platform a *version* definující platformu, na které zařízení běží. Pole *version* není povinné, ale je doporučené.

```
[type] => string
[udid] => string
[platform] => ios/android/win
[version] => string
```

Po úspěšném dekódování a zkontrolování je dle pole *type* vykonána odpověď. Odpověď je sestavena do asociativního pole v PHP, ze kterého se generuje JSON struktura pomocí funkce *json_encode($array)* a nastavení HTTP hlavičky na *content-type: application/json*.

```
/* Return as JSON array */
header("Content-type: application/json");
echo json_encode($out);
```

Pokud byly parametry požadavku zadány správně, tak odpověď bude v následujícím formátu:

```
[status] => true
[response]=> array/object
```

V případě špatně zadaných parametrů bude vyhozena výjimka s chybou v následujícím formátu:

```
[status] => false
[error] => string
[code] => int
```

Systém Tagiee odpovídá na následující typy požadavků. Pro ukázku bude uveden příklad jednoho dotazu nad API. Celá dokumentace API je v příloze.

get_institution_list

Vrací seznam institucí do vzdálenosti 100 km. Předpokladem je určení zeměpisné polohy uživatele v požadavku. Kromě povinných parametrů požadavku je navíc potřeba vyplnit tyto pole:

```
/* Request */
[type] => get_institution_list
[lat] => double
[lon] => double
```

```
/* Response */
[response]=>
    (
        [1..N array]
            [revision] = int
            [logo] => string
            [uiid] => string
            [name] => string
            [address] => string
            [lat] => double
            [lon] => double
            [rating] => double
            [number_reviews] => int
    )
```

V odpovědi je seznam institucí se jménem, adresou, unikátního identifikátoru instituce *uiid*, zeměpisné polohy a hodnocení instituce uživateli.

get_institution_by_uiid

Na základě platného unikátního identifikátoru vrací textové informace o instituci.

Informace o instituci jako je název, adresa, zeměpisné souřadnice, otevírací doby, hodnocení uživatelů, ceník, kontaktní informace. Seznam všech majáků spolu s jejich textovým obsahem a seznam všech použitých zdrojů v obsahu majáků.

get_institution_revision

Na základě unikátního identifikátoru instituce vrátí aktuální revizi instituce. Aplikace si může díky tomuto požadavku zjistit, jestli má nejnovější verzi dat.

device_visit_institution

V případě přihlášení k instituci je poslána notifikace systému Tagiee, která uchová tuto informaci pro budoucí statistiky návštěvnosti.

beacon_found

Pokud návštěvník nalezene maják, pošle systému Tagiee notifikaci informující o tomto nálezu, též pro budoucí statistiky návštěvnosti majáků.

get_review

Vrátí seznam posledních hodnocení instituce. Díky požadavku *get_institution_by_uiid* mobilní zařízení zjistí počet hodnocení a na základě tohoto počtu si může aplikace sama určit, kolik recenzí chce vrátit a hodnocení stránkovat.

add_review

Umožní přidání hodnocení uživatelem. Jeden uživatel může přidat pouze jedno hodnocení k dané instituci.

admin

Na základě přihlašovacích údajů poskytne zaměstnanci instituce informace o svých přidaných institucích a majácích.

add_location

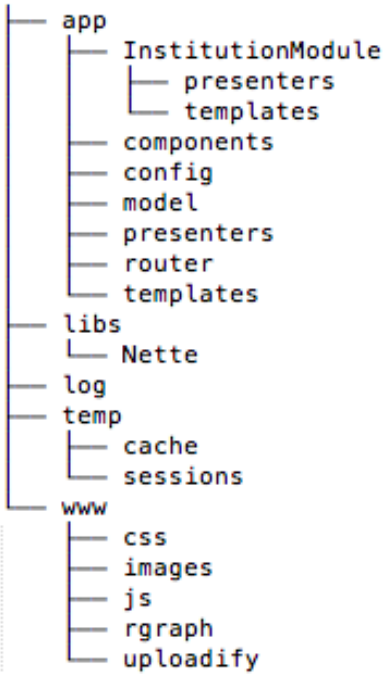
Přihlášení uživatelé můžou ke svým institucím přidat lokace, na kterých se nacházejí. Přidání není povoleno z důvodu bezpečnosti, pokud nejsou předány přihlašovací údaje správce instituce.

4.5 Implementace webového rozhraní

K implementaci webového prostředí systému Tagiee jsem využil PHP a framework Nette. Webové prostředí bude sloužit zaměstnancům institucí pro vytváření a editování obsahu o jejich instituci.

4.5.1 Struktura

Struktura webové části (Obr. 4.3) Tagiee je rozdělena na 2 funkční moduly. Prvním modulem je defaultní prostředí pro přihlášení, registraci, přidání instituce a druhý modulem je samotné editování obsahu instituce. První modul se nachází ve složce app, respektive jeho presentery a šablony ve složkách app/presenters a app/templates. Druhý modul je v podsložce app/InstitutionModule a presentery, šablony ve složkách app/InstitutionModule/presenter a app/InstitutionModule/templates.



Obrázek 4.3: Struktura webového rozhraní.

Datová vrstva je společná pro oba moduly a je umístěna ve složce app/-model. O datové vrstvě (viz 4.5.2).

Ve složce www/ jsou umístěny CSS styly, Javascript, obrázky související s grafikou webu a podpůrné soubory knihoven.

4.5.2 Databázová vrstva

Základem databázové vrstvy je třída Repository. Konstruktorem se předá instance Nette/Database/Connection, která umožňuje připojení k databázi. Konkrétní připojení je definované ve složce app/config/config.neon následujícím způsobem:

```
database:
        dsn: 'mysql:host=127.0.0.1;dbname=tagieecom1'
        user: 'tagiee.com'
        password: 'XXXXXX'
```

Třída Repository má důležitou metodu *getTable()*. Tato metoda na základě jména třídy vybere tabulku, ze které bude číst záznamy. Pravidlo pro jmenování tříd je jméno tabulky začínající velikým písmenem následující slovem Repository.

Pokud tedy chceme vrátit všechny uživatele systému Tagiee, zavoláme metodu *getTable()* nad instancí třídy *UserRepository*. Třída UserRepository je definovaná následovně.

```
class UserRepository extends Repository{

}
```

Třída *UserRepository* je bez metod, protože si plně vystačí s třídami svého rodiče. Metoda *getTable()* na základě názvu třídy bude vybírat záznamy z tabulky *user*. Pokud bychom si nevystačili s metodami rodiče, není problém rozšířit tuto třídu o své metody. Rozdělení na podtřídy přidává na přehlednosti datové vrstvy a jasně definuje i její chování.

Třída *Repository* obsahuje tyto metody pro práci s tabulkami:

- *getTable()* - Vrátí objekt reprezentující databázovou tabulku. Pokud jinak nespecifikujeme výběr, vrací všechny záznamy v tabulce jako Nette/Database/Table/Selection, jímž se dá procházet jako polem.
- *findBy(array $condition)* - Najde specifický záznam v tabulce, parametrem je podmínka jako asociativní pole, tedy název_sloupečku => hodnota. Vrací se jako objekt Nette/Database/Table/Selection.
- *insert(array $values)* - Vloží nový záznam do tabulky, parametrem je opět asociativní pole název_sloupečku => hodnota.

- *update(array $condition, array $values)* - Aktualizuje záznam v tabulce. Parametrem jsou dvě asociativní pole. První je podmínka, jaký záznam se bude aktualizovat a druhým parametrem jsou hodnoty, které se mají aktualizovat.
- *delete(array $condition)* - Smaže záznam z tabulky. Parametrem je podmínka, která vybere záznam, jenž se smaže.
- *exec($query, $data)* - Vykoná nad databází SQL příkaz. Prvním parametrem je příkaz obsahující jeden nebo více znaků „?“ a druhým parametrem je pole hodnot, které se dosadí místo otazníků.

Aby byly třídy modelů jednodušeji použitelné, zaregistrují se jako služby, tudíž se presenter nemusí starat o jejich vytváření. Jako služby se zaregistrují do souboru app/config/config.neon následujícím způsobem:

```
services:
        userRepository: DB\UserRepository
        ...
        institutionRepository: DB\InstitutionRepository
```

Název na levé straně před dvojtečkou definuje instanci *UserRepository* jako službu, kterou je možné použít kdykoliv v presenterech. Pravá strana určuje, jaká instance bude vytvořena. Objekt *UserRepository* dědící od třídy *Repository* však vyžaduje při vytvoření instance v konstruktoru instanci třídy *Connection*. Avšak Nette má funkci Auto-Wiring, která pozná, že je třeba instance *Connection* a tuto instanci za programátora vytvoří.

Pro úplnost bude ukázáno použití těchto modelových tříd v presenterech:

```
class RegisterPresenter extends BasePresenter {

        /** @var DB\UserRepository **/
        private $userRepository;


        protected function startup() {
                parent::startup();
                // Ziskani instance DB\UserRepository
                $this->userRepository = $this->context->
                        userRepository;
        }


        public function registerFormSucceeded($form) {
```

```
                ....
                $data = array('username' => $values->username,
            'password' => Authenticator::calculateHash(
                        $values->password),
            .....
            'post_code' => $values->post_code);
                // Pridani uzivatele do tabulky
                $this->userRepository->insert($data);
        }


}
```

4.5.3 Správa uživatelů a institucí

Po přistoupení na adresu http://tagiee.com se zobrazí úvodní přihlašovací obrazovka s informacemi o službě Tagiee, k čemu slouží a pro koho je určena. Přihlásit se půjde pod účtem, který si zaměstnanec vytvoří. Na úvodní stránce je také odkaz k registraci nového uživatele.

Na celém webu je dostupná horní lišta, správa přihlášeného uživatele a možnost jeho odhlášení.

Na adrese http://tagiee.com/register/ je registrační formulář nového uživatele. Po uživateli je požadováno kromě uživatelského jména a hesla detailní kontaktní údaje. Detailní informace jsou potřeba z důvodu, aby se neregistroval každý a aby obsah služby Tagiee byl hodnotný.

K přihlášení se využívá třída Authenticator a její statická metoda *calculateHash($heslo, $sůl = null)*, která pomocí PHP funkce *crypt($heslo, $sůl = null)* vytvoří z hesla „osolený“ zakódovaný řetězec, který se při přihlášení porovnává se zadaným heslem. Pro vytvoření se zakóduje řetězec pomocí této metody:

```
/* Zakodovani hesla */
Authenticator::calculateHash($form->values->password);
```

Při přihlášení se pak porovná zakódovaný řetězec z databáze s touto metodou. Metoda pak vrátí úplně stejný zakódovaný řetězec. Tím dojde k porovnání a ověření, že se heslo shoduje.

```
/* Heslo k overeni. */
$password = $form->values->password;
```

```
/* Ziskani zakodovaneho retezce z databaze. */
$row = $this->database->table('user')
        ->where('username', $username)->fetch();

if($row->password !==
        $this->calculateHash($password, $row->password)) {
        /* Hesla se neshoduji. */
}
```

Pokud je uživatel přihlášen, na úvodní stránce se mu zobrazí seznam jeho institucí a možnost jejich přidání. Pro přidání instituce je nutné zadat její název, unikátní identifikátor a informace o umístění. Pokud chce zaměstnanec smazat instituci, je nutné kontaktovat správce systému Tagiee z bezpečnostních důvodů a nechtěného smazání instituce. V případě potřeby je možné deaktivovat instituci při jejím editování (viz 4.5.4). Z úvodní obrazovky lze vybráním instituce přejít k její editaci.

4.5.4 Vytváření a editování obsahu institucí

Editace instituce je dostupná na adrese http://tagiee.com/institution/<uiid>. Editování instituce je jen pro přihlášené uživatele. Každý uživatel si může editovat jen svoje vytvořené instituce.

Editování instituce je rozděleno na několik částí, které mají svou specifickou adresu a ke kterým se lze dostat z levé postranní nabídky.

Informace a statistiky

Tato stránka se zobrazí pod adresou http://tagiee.com/institution/<uiid>. Lze zde aktivovat / deaktivovat instituci. Deaktivovaná instituce nebude dostupná skrze klientskou aplikaci. Nově vytvořená instituce není aktivovaná a lze ji aktivovat pouze, pokud obsahuje název, popisek, logo instituce, kontaktní telefon a email, otevírací dobu, ceník, zeměpisné souřadnice a obsahuje alespoň jeden aktivní maják.

Na stránce lze nalézt různé statistiky týkající se instituce a jejích majáků, počet přístupů, nalezení majáků dle technologií a podíl zařízení s operačním systém a jejich jednotlivých verzí. K zobrazení jsou využity tabulky a ko-

láčové grafy pomocí Google Charts (viz 4.2.5). V případě statistik jsou dotazy nad databází sestavovány ručně a kvůli rychlosti se nevyužívá NotORM Nette.

Úprava instituce

Úprava údajů o instituci využívá adresu http://tagiee.com/institution/<uiid>/edit/. Na úpravu se lze dostat i z levé postranní nabídky. Všechny informace o instituci jsou měnitelné kromě unikátního identifikátoru uiid instituce. Logo se nahrává skrze knihovnu Uploadify (viz 4.2.8). Logo musí být obrázek ve formátu PNG, JPG či JPEG, musí mít minimální rozlišení 250x250px a velikost menší než 300 Kb. Minimální rozlišení je pro zachování kvality obsahu. Omezení velikosti obrázku je z důvodu snížení přenosu dat mobilní datovou sítí. Nahraný obrázek je oříznut do čtverce.

Po nahrání je možno tento obrázek upravit pomocí knihovny Jcrop (viz 4.2.9) a zvolit si jinou čtvercovou výseč v obrázku.

Editace údajů o instituci se provádí bez nutnosti znovu načtení stránky.

Přidání, úprava majáků

Přidání majáků má adresu http://tagiee.com/institution/<uiid>/beacons. Zde je umístěn seznam majáků, jejich přehled a je možno kdykoliv majáky smazat. Pro přidání stačí název majáku. Nově přidaný maják je neaktivní. Novému majáku se vygeneruje unikátní identifikátor UBID, který se skládá z unikátního identifikátoru instituce a názvu majáku. Dále se novému majáku vygeneruje náhodný unikátní 3-místný klíč, jenž je unikátní u všech majáků v instituci. Pomocí tohoto 3-místného klíče lze pak také nalézt maják klientskou aplikací.

Úprava majáků má vlastní stránku, na maják se lze dostat i pomocí adresy http://tagiee.com/institution/<uiid>/beacons/<ubid>/. Pro úpravu majáků není nutnost znovu načtení stránky, komunikace probíhá AJAXovými požadavky. Neaktivovaný maják lze aktivovat pouze, pokud je vyplněno jméno, popisek a vytvořen obsah, který se zobrazí po jeho nalezení. Nesplní-li se tyto podmínky, formulář pro aktivaci majáku nebude zobrazen.

Obsah majáku podporuje přidání textů, odkazů, obrázků a jednoho You-

tube videa. Obsah majáků je uložen do databáze jako JSON pole a je omezen velikostí sloupečku content v tabulce beacon, což je 15000 znaků.

Přidání textu a odkazu se provádí přetažením elementu do kontejneru, ve kterém se můžou libovolně řadit. K tomu se používá knihovna JQuery, respektive JQuery UI. Nad kontejneru je plugin *.sortable()*, který řadí elementy v kontejneru a nad elementy je plugin *.draggable()*, který umožní jejich přetažení do kontejneru.

```
    /* Plugin pro text a odkaz */
$( "#text, #href" ).draggable({
  connectToSortable: "#sortable",
  helper: "clone",
  revert: "invalid"
});

/* Plugin pro kontainer */
$( "#sortable" ).sortable();
```

Po kliknutí na elementy v kontejneru se zobrazí dialogové okno, které umožní jejich editaci. Pro zobrazení dialogu se využívá plugin dialog z knihovny JQuery UI.

Elementy pro přidání obrázků a videa se nevkládají do kontejneru. Jsou samostatné a umožňují přidání až deseti obrázků a jednoho videa.

Poloha majáků se udává pomocí lokací, které se definují v samotné sekci.

Lokace

Lokace jsou dostupné na adrese http://tagiee.com/institution/<uiid>/location. Lokace se používají pro definování polohy majáků při jejich úpravě. Lokace je zobrazena pomocí komponenty LocationControl. Jednotlivé zobrazené komponenty LocationControl musí mít vlastní instanci a být navzájem nezávislé. K tomu se využívá třída Multiplier, která pro každou komponentu vytvoří vlastní instanci. Díky této třídě je možné mít více nezávislých komponent na jedné stránce.

Nahrání obrázků

Obrázky jsou na adrese http://tagiee.com/institution/<uiid>/resource/ a vkládají se pomocí knihovny Uploadify (viz 4.2.8). Obrázek musí mít minimální rozlišení 250x250px a velikost menší než 2 Mb. Velikost je omezena kvůli hardwarově slabším mobilním zařízení (některá zařízení se systémem Android nedokáží zobrazit obrázek větší než 2 Mb).

5 Klientská část

Klientská část byla implementována pro platformu iOS (viz 5.1). V této kapitole vycházím ze své bakalářské práce, kde jsem se zabýval vývojem aplikací pro platformu iOS v jazyce Objective-C 2.0 a popisoval vlastnosti jazyka a principy vývoje mobilních aplikací.

Realizace klientské části Tagiee byla vyvíjena na operačním systému OS X 10.8.3 ve vývojovém prostředí Xcode 4.6.1. a bude dostupná pro platformu iOS 6 a novější. Minimální verze 6.0 byla vybrána díky novinkám ve vývoji aplikací a tomu, že 81,2% zařízení iOS běží ve verzi 6 a více (viz 5.2).

5.1 Platforma iOS

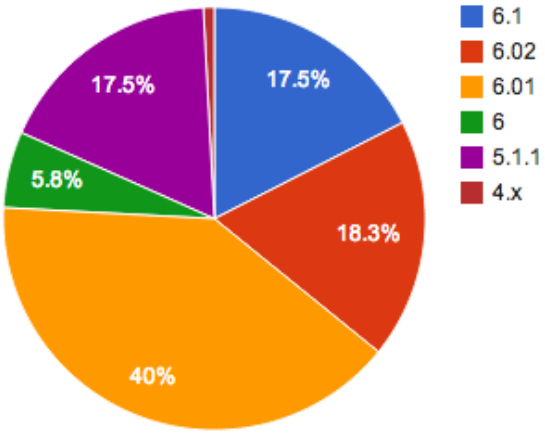
Mobilní operační systém iOS byl vytvořen firmou Apple Inc. Původně byl určen pro mobilní zařízení iPhone. Nyní je však operační systém využíván i v zařízeních jako iPod Touch, iPad a Apple TV. [ios()]

Systém iOS je odvozen z operačního systému OS X. Tyto dva operační systémy spolu sdílejí OS X kernel, BSD sokety pro síťovou komunikaci, jazyk Objective-C a C/C++ kompilery. OS X pro tvorbu grafických interaktivních aplikací využívá framework Cocoa, který se skládá z knihoven a vysokoúrovňového API. Aplikace, vytvořené nad tímto frameworkem, získají vzhled a funkčnost nativních aplikací systému OS X. Cocoa využívá architekturu MVC[1]. Systém iOS využívá framework Cocoa Touch, který je uzpůsoben pro dotyková zařízení. [ios()]

5.2 Podíl verzí iOS na trhu

Dle výzkumu společnosti 14 Oranges Software Inc. je podíl zařízení iOS 6 dominantní na trhu (viz Obr. 5.1) k 14. únoru 2013. [ora()]

[1]Model-View-Controller rozděluje datovou vrstvu, řídící logiku a uživatelské rozhraní do tří nezávislých komponent.



Obrázek 5.1: Podíl verzí iOS na trhu.

5.3 Použité knihovny

Všechny uvedené knihovny jsou volně dostupné a šiřitelné. Na knihovnách je ukázán způsob jejich základního použití. V implementační části aplikace (viz 5.4) je popsána jejich úprava pro použití v aplikaci.

5.3.1 AFNetworking

AFNetworking je síťová knihovna pro iOS a OS X postavená na frameworku Foundation, tedy hlavně na NSURLConnection a NSOperation. Knihovna dovoluje posílat HTTP požadavky nebo stahovat obrázky případně soubory. Téměř všechny metody knihovny využívají bloky kódů, které se zavolají při úspěšném / neúspěšném HTTP požadavku. Knihovna umožňuje vytvoření HTTP klienta, který může být nastaven pro komunikaci s webovými službami. HTTP klient využívá frontu požadavků, tudíž např. při stahování galerie obrázků se obrázky stáhnou postupně a nevytíží tolik datovou linku. Na následujícím příkladu je použití knihovny AFNetworking pro komunikaci s webovým API, které vrátí odpověď ve formě JSON struktury. [afn()]

```
/* Adresa pro komunikaci s API */
NSURL *url = [NSURL URLWithString:@"https://api.net/posts/0/"];
```

```
/* Vytvoreni HTTP pozadavku. */
NSURLRequest *request = [NSURLRequest requestWithURL:url];
AFJSONRequestOperation *operation = [AFJSONRequestOperation
        JSONRequestOperationWithRequest:request
                success:^(NSURLRequest *request,
                NSHTTPURLResponse *response,
                        id JSON) {
                                /* Komunikace uspesna. */
                                } failure:nil];
/* Zahajeni komunikace. */
[operation start];
```

5.3.2 ZBarSDK

ZBarSDK je volně šiřitelná knihovna, která umožňuje v reálném čase čtení kódů EAN-13/UPC-A, UPC-E, EAN-8, Code 128, Code 39, Interleaved 2 of 5 a QR kódů. Pro použití stačí implementovat metody protokolu *ZBarReaderDelegate* a vytvořit instanci *ZBarReaderViewController*. Pokud je vše připravené, je možné zobrazit Zbar čtečku například jako modální okno. [zbd()]

5.3.3 OpenUDID

Od verze iOS 5.0 Apple zakázal použití systémového unikátního identifikátoru zařízení takzvané UDID. Všechny nové aplikace využívající tohoto identifikátoru budou zamítnuty k publikování do obchodu App Store. Stávající aplikace využívající UDID budou do konce května 2013 vyřazeny z obchodu.

Proto vznikla volná implementace UDID, která může unikátně identifikovat zařízení napříč mobilními platformami. K identifikaci se využívá sériové číslo procesoru a MAC adresa WiFi adaptéru. Použití je následující:

```
#include "OpenUDID.h"
NSString* openUDID = [OpenUDID value];
```

Statická metoda *value* vrací unikátní řetězec zařízení o délce 40ti znaků. Příklad takového řetězce je „1c2cf41ab0e5c4a83702ee0c7e77c11d2cdfb3d6“. [ope()]

5.3.4 MWPhotoBrowser

MWPhotoBrowser (viz Obr. 5.2) je volná implementace nativní obrázkové galerie systému iOS. Snaha knihovny je být přesnou kopií nativní galerie. Knihovna zvládá cachování a stahování obrázků v průběhu prohlížení. [mwp()]

Obrázek 5.2: Ukázka MWPhoto galerie.

Implementace je pomocí modálního okna nebo pomocí navigačního controlleru.

5.3.5 DYRateView

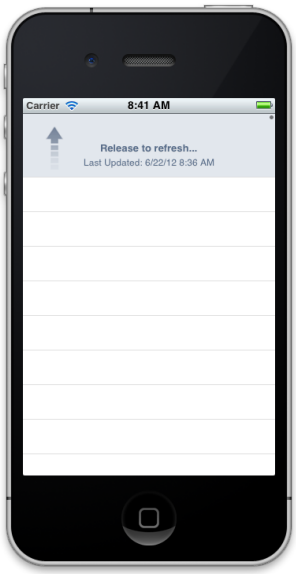
DYRateView je univerzální komponenta, která zobrazuje hodnocení ve formě pěti hvězdiček. Komponenta může být i editovatelná a sloužit jako zdroj hodnocení od uživatelů aplikace. [dyr()] DYRateView je objekt UIView, použití je tedy následující:

```
DYRateView *rateView = [[DYRateView alloc] initWithFrame:frame];
rateView.rate = 4.7;
rateView.alignment = RateViewAlignmentRight;
[self.view addSubview:rateView];
```

5.3.6 EGOTableViewPullRefresh

EGOTableViewPullRefresh (viz Obr. 5.3) je rozšíření UITableView o možnost stáhnout prstem tabulku a znovu načíst její obsah. EGOTableView-

PullRefresh se přidá do tabulky jako její hlavička a implementováním scrollView protokolu tabulky se vloží funkčnost. [ego()]



Obrázek 5.3: Znovu načtení tabulky stáhnutím prstem.

5.3.7 ECGraph

ECGraph je knihovna, která vykreslí v jakémkoliv UIView graf. Knihovna podporuje kreslení sloupcových a spojnicových grafů. Na následujícím příkladu je implementace sloupcového grafu. [ecg()]

```
CGContextRef _context = UIGraphicsGetCurrentContext();
ECGraph *graph = [[ECGraph alloc] initWithFrame:
                CGRectMake(10,10, 480, 320)
                withContext:_context isPortrait:NO];
NSArray *items = [[NSArray alloc] initWithObjects:item1,item2
                                        ,nil];
[graph setXaxisTitle:@"name"];
[graph setYaxisTitle:@"Percentage"];
[graph setGraphicTitle:@"Histogram"];
[graph setDelegate:self];
[graph drawHistogramWithItems:items lineWidth:2
                color:[UIColor blackColor]];
```

Pro vykreslení se využívá funkce *UIGraphicsGetCurrentContext()*, která slouží pro kreslení uvnitř UIView. Předpokladem je volání této funkce v me-

todě *drawRect()* každého UIView, který chce implementovat graf.

5.4 Implementace klientské aplikace

Klientská aplikace byla implementována na základě návrhu (viz 3.3.2).

5.4.1 Komunikace se systémem Tagiee

Ke komunikaci s klientským API byla využita knihovna AFNetworking (viz 5.3.1).

Základem je společný HTTP klient, který je singletonem a má společnou instanci pro komunikaci s API v celé aplikaci. Klient je již přednastaven pro komunikaci pomocí JSON struktur. Komunikace nad tímto klientem se provádí postupně a tudíž nedochází k zahlcení sítě.

```
static NSString * const kAPIURL = @"http://api.tagiee.com";
@implementation TagieeClient

+ (TagieeClient *)sharedClient {
    static TagieeClient *_sharedClient = nil;
    static dispatch_once_t onceToken;
    dispatch_once(&onceToken, ^{
        _sharedClient = [[TagieeClient alloc] initWithBaseURL:
        [NSURL URLWithString:kAPIURL]];
    });

    return _sharedClient;
}

- (id)initWithBaseURL:(NSURL *)url {
    self = [super initWithBaseURL:url];
    if (!self) {
        return nil;
    }

    // Response type JSON
    [self registerHTTPOperationClass:
                [AFJSONRequestOperation class]];
    // Request type JSON
    [self setParameterEncoding:AFJSONParameterEncoding];
```

```
    // Accept HTTP Header;
        [self setDefaultHeader:@"Accept"
                  value:@"application/json"];

    return self;
}

@end
```

Takto přednastavený klient je již schopný komunikovat s Tagiee Web API (viz 4.4) pomocí definovaných požadavků. Použití klienta pro přístup k instituci vypadá následovně:

```
 NSDictionary *jsonParams = @{
        @"type" : @"get_institution_by_uiid",
        @"uiid":uiid,
        @"udid":[OpenUDID value],@"platform":@"ios",
        @"version":[[UIDevice currentDevice] systemVersion]};

[[TagieeClient sharedClient] postPath:@"" parameters:jsonParams
success:^(AFHTTPRequestOperation *operation, id JSON) {
        NSDictionary *response = JSON;
        // Komunikace uspesna, zpracovani instituce
} failure:^(AFHTTPRequestOperation *operation, NSError *error) {
        // Doslo k chybe pro komunikaci
}
```

Požadavek i odpověď jsou ve formě objektu NSDictionary, tedy asociativního pole. Pro zjednodušení jsou jednotlivé typy požadavků Web API ve vlastních třídách, které definují protokolové metody, které se zavolají v případě úspěchu či neúspěchu. Všechny typy požadavků systému Tagiee byly implementovány.

5.4.2 Databázová vrstva

Framework Core Data

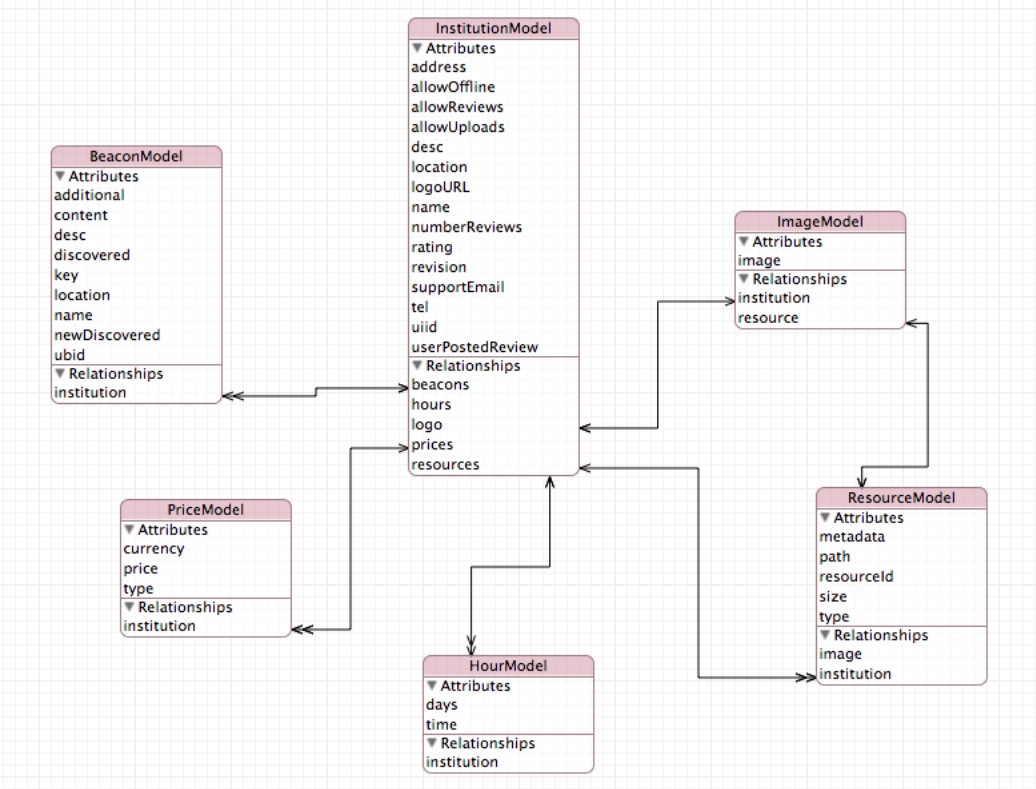
Framework Core Data je persistentní úložiště, které se definuje entitami a vazbami mezi nimi. Framework má v sobě podporu funkcí pro editování textu, lazy loading[2], vyhledávání v datech, migraci dat a správu vazeb mezi

[2]Návrhový vzor načítání objektů z databáze až když jsou potřeba. Zvyšuje se výkon čtení dat a snižují se paměťové nároky.

entitami. Data se mohou uložit do souboru XML nebo jako SQLite soubor.

Datový model

Datový model v aplikaci (viz Obr. 5.4) byl vytvořen pomocí editoru Xcode. Model v sobě ukládá informace o instituci a jejích majících. Stáhnuté obrázky jsou také uloženy v databázi a to jako BLOBs[3] data. Apple uvádí, že s ukládáním obrázků do Core Data se musí opatrně. Dá se dosáhnout dobrých výkonnostních výsledků s daty do 10 MB, s tím že BLOBs budou uloženy v samotné entitě a bude na ně vazba. Tím, že Core Data používá lazy loading, bude obrázek načten až v případě jeho potřeby.



Obrázek 5.4: Datový model Core Data v aplikaci.

[3]Binary Large OBjects - Veliká nadměrná data například obrázky nebo zvuky.

Použití Core Data v aplikaci

V aplikaci je použito úložiště SQLite pro Core Data. Pomocí Xcode se vygeneruje objektový model, který odpovídá datovému modelu. Každá entita bude reprezentovaná třídou oděděnou od NSManagedObject. Vazby mezi entitami budou propojeny pomocí pole, pokud se jedná o vazbu 1:N nebo o referenci na jinou třídu vazby 1:1.

Nejprve se nahraje objektový model do paměti.

```
- (NSManagedObjectModel *)managedObjectModel {
    if (_managedObjectModel != nil) {
        return _managedObjectModel;
    }
    _managedObjectModel = [NSManagedObjectModel
        mergedModelFromBundles:nil];
    return _managedObjectModel;
}
```

Pokud je načtený objektový model, získá se koordinátor na Core Data úložiště. V tomto případě na soubor TagieeModel.sqlite. Koordinátor slouží jako prostředník mezi objektovým modelem a tím datovým, tudíž mapuje objekty na jednotlivé entity.

```
- (NSPersistentStoreCoordinator *)persistentStoreCoordinator
{
    if (_persistentStoreCoordinator != nil) {
        return _persistentStoreCoordinator;
    }

    NSURL *storeURL = [[self applicationDocumentsDirectory]
        URLByAppendingPathComponent:@"TagieeModel.sqlite"];

    NSError *error = nil;
    _persistentStoreCoordinator =
        [[NSPersistentStoreCoordinator alloc]
    initWithManagedObjectModel:[self managedObjectModel]];
    if (![_persistentStoreCoordinator addPersistentStoreWithType:
                NSSQLiteStoreType configuration:nil URL:storeURL
                options:nil error:&error]) {
                // Nelze pouzit model
    }

    return _persistentStoreCoordinator;
}
```

Pro koordinátor je nutné vytvořit prostor nebo-li kontext, ve kterém se

bude pracovat s daty (číst je a měnit). To zajistí objekt NSManagedObject-Context, kterému se předá koordinátor, podle kterého bude vědět, jaký typ dat si má uchovat a jaké informace poskytnout.

```
- (NSManagedObjectContext *)managedObjectContext
{
    if (_managedObjectContext != nil) {
        return _managedObjectContext;
    }

    NSPersistentStoreCoordinator *coordinator = [self
                                persistentStoreCoordinator];
    if (coordinator != nil) {
        _managedObjectContext = [[NSManagedObjectContext alloc]
                                                    init];
        [_managedObjectContext setPersistentStoreCoordinator:
                                                    coordinator];
    }
    return _managedObjectContext;
}
```

Čtení dat z kontextu se pak provádí pomocí objektu NSFetchRequest, který na základě formálního popisu objektu NSEntityDescriptor vrátí například všechny instituce uložené v Core Data.

```
NSFetchRequest *fetchRequest = [[NSFetchRequest alloc] init];
NSEntityDescription *entity = [NSEntityDescription
            entityForName:@"InstitutionModel"
            inManagedObjectContext:context];
[fetchRequest setEntity:entity];
NSError *error;

for (InstitutionModel* institution in
[context executeFetchRequest:fetchRequest error:&error]){
        // Instituce
}
```

Tímto se dostane objekt InstitutionModel, ve kterém jsou data o instituci. Tyto data se můžou měnit a následně pak uložit. Uložení se provede metodou *save:* nad kontextem. Aby se uložení projevilo, je nutné aby se jednalo o stejnou instanci třídy InstitutionModel, jaká byla vrácena z objektu NSFetch-Request. Pro ukázku bude uvedeno přidání nové instituce do Core Data pomocí metody *insertNewObjectForEntityForName:inManagedObjectContext:* nad třídou NSEntityDescription.

```
InstitutionModel *institution = [NSEntityDescription
        insertNewObjectForEntityForName:@"InstitutionModel"
        inManagedObjectContext:context];
```

5.4.3 Uživatelské rozhraní

Tato část se bude zabývat popisem uživatelského rozhraní jen pro iPhone a iPod Touch. Aplikace se skládá z několika obrazovek (viz Obr. 5.5), kde základním kamenem je UINavigationViewController. UINavigationViewController je komponenta, která umí zobrazit obsah (UIViewControllery) v hierarchické struktuře. Základním prvkem UINavigationViewController je navigační lišta, která obsahuje titulek popisující, kde se uživatel právě nachází a případné tlačítko zpět, které umí vrátit na předchozí zobrazenou obrazovku.

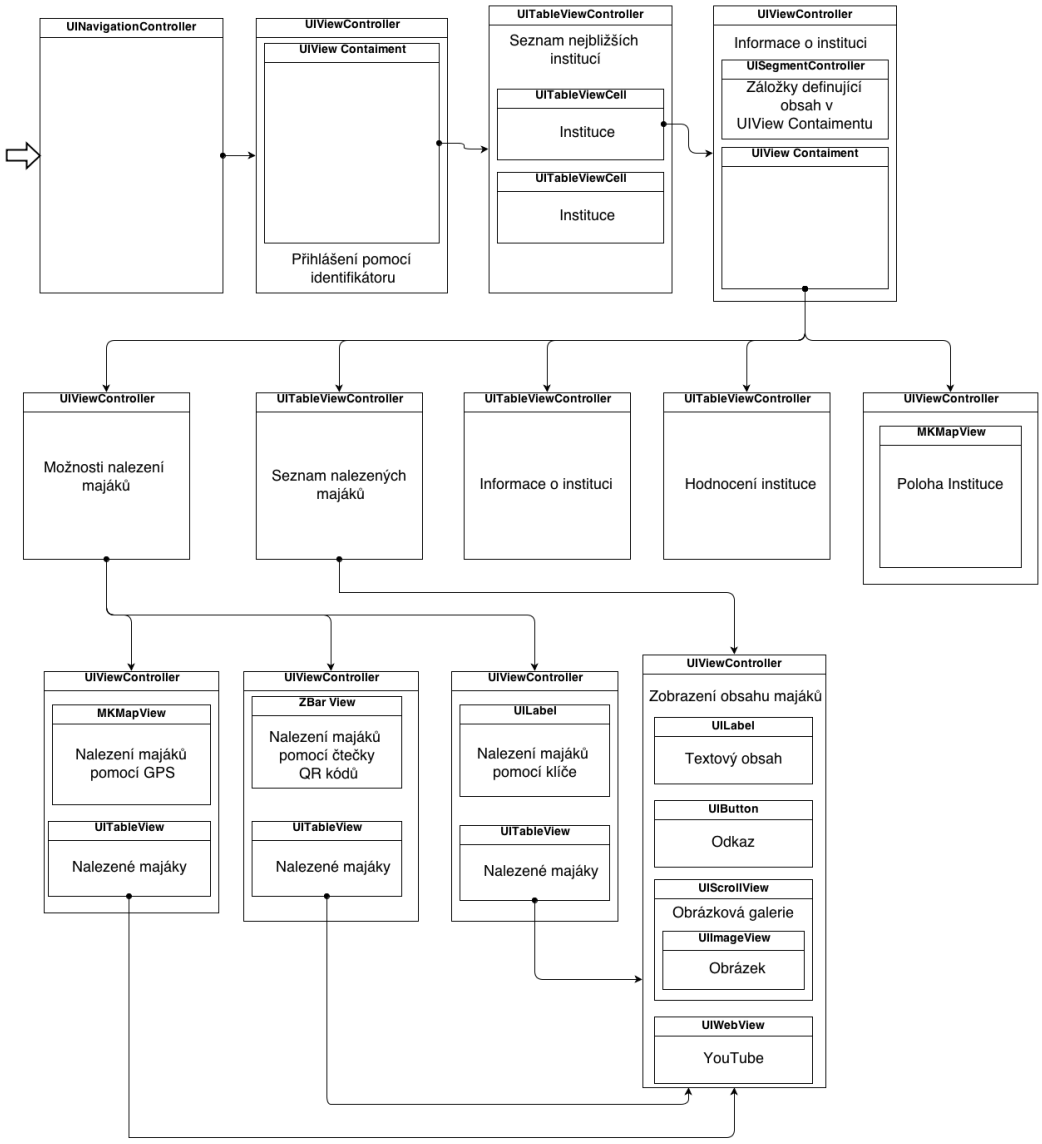
Úvodní obrazovka

Po zapnutí aplikace se uživateli zobrazí nejbližší dostupné instituce (viz Obr. 5.6). Nejbližší instituce jsou zobrazeny pomocí UITableViewControlleru, který je rozšířen o EGOTableViewPullRefresh (viz 5.3.6) aktualizaci obsahu posunutím prstu směrem dolů. Seznam nejbližších institucí je zobrazen jako UIView containment, jenž umožňuje vložení UIViewControlleru do jiného. Tím je možné na obrazovce zobrazit spodní tlačítko spolu se seznamem nejbližších institucí. Uživatel se může přihlásit k instituci i pomocí unikátního identifikátoru instituce dotykem na spodní tlačítko. Pomocí tlačítek v navigační liště se může uživatel dostat do administrace a historie.

Zobrazení instituce

Po přihlášení k instituci se zobrazí obrazovka (viz Obr. 5.7), ve které je možné přistoupit k nalezení majáků, k informacím o instituci, k jednou již nalezeným majákům, zobrazit hodnocení uživatelů a přesnou polohu instituce. Uprostřed se nachází UISegmentController, tedy záložky, které definují obsah pod ním. V případě že obsah, který se nachází pod záložkami, bude dlouhý a nevejde se na obrazovku, je možno jej posunout díky UIScrollView. Při posunu se záložky připnou k hornímu části obrazovky, tudíž jsou vždy vidět.

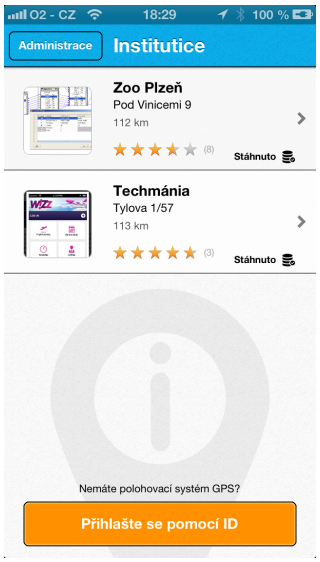
Pod záložkami je obsah opět pomocí UIView containment. Tedy schránka, která může v sobě zobrazovat další UIViewControllery (informace o instituci, nalezené majáky, hodnocení uživatelů,...). Schránka má svého rodiče, tedy UIViewController, který zobrazuje hlavičku s logem instituce a záložky. Mezi těmito UIViewControllery se přepíná následujícím způsobem.



Obrázek 5.5: Uživatelské rozhraní mobilní aplikace.

```
- (void) cycleFromViewController: (UIViewController*) oldC
                toViewController: (UIViewController*) newC{
    /* Odebere se stary UIViewController. */
    [oldC willMoveToParentViewController:nil];
    [oldC.view removeFromSuperview];
    [oldC removeFromParentViewController];


    /* Prida se novy UIViewController. */
```



Obrázek 5.6: Úvodní obrazovka aplikace.

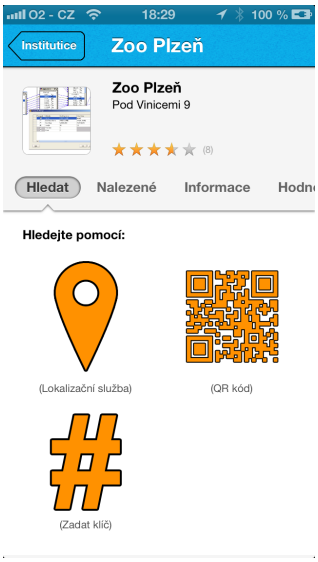
```
    [self addChildViewController:newC];
    newC.view.frame = _container.bounds;
    [_container addSubview:newC.view];
    [newC didMoveToParentViewController:self];
}
```

Bohužel tím, jak jsou UIViewControllery ve schránce (berou všechny doteky rodičovského UIViewControlleru a zpracovávají si je samy), je problém dosáhnout posouvání celé obrazovky. Trikem je přepočíst velikost obsahu (UIViewControlleru ve schránce), a nastavit tak schránce požadovanou velikost. UIScrollView se bohužel také neumí sám od sebe se roztahovat dle obsahu, proto je nutné i jemu nastavit velikost.

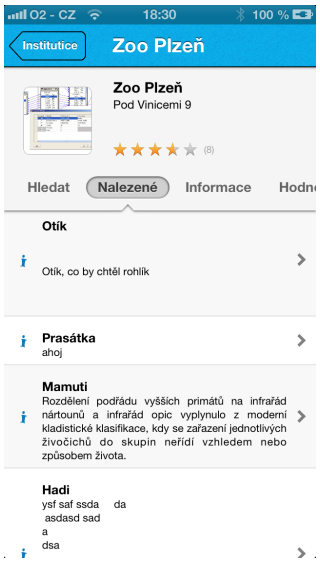
```
- (void)updateScrollViewWithView:(UIView*)view{
    /* Vypocte se velikost noveho obsahu. */
    CGSize size = [view sizeThatFits:
        CGSizeMake(_container.frame.size.width, CGFLOAT_MAX)];

    /* Spocte se celkova velikost i s hlavickou a schrankou */
    CGFloat height = _container.frame.origin.y +
                        view.frame.size.height;

        /* Pokud je velikost mensi nez cela obrazovka,
        nastavi se velikost + 1 */
    if(height <= _scrollView.frame.size.height)
        height = _scrollView.frame.size.height + 1;
```

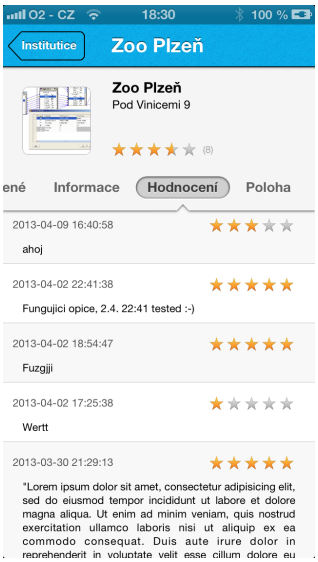


(a) Obrazovka pro nalezení majáků.



(b) Nalezené majáky.

(c) Informace o instituci.



(d) Obrazovka s hodnocením uživatelů.

(e) Zobrazení polohy instituce.

Obrázek 5.7: Obrazovka zobrazující informace o instituci.

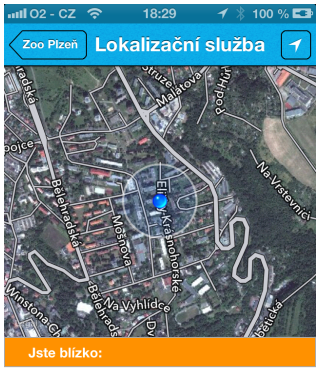
```
/* Upravy se velikost  UIScrollView */
_scrollView.contentSize =
    CGSizeMake(_scrollView.frame.size.width, height);
/* Upravy se velikost containeru */
```

```
    CGRect frame =  _container.frame;
    frame.size.height = height;
    _container.frame = frame;
}
```

Změní-li se ve schránce UIViewController, tak nový UIViewController zavolá nad svým rodičem metodu *updateScrollViewWithView:*, která zjistí velikost jeho hlavního UIView podle metody *sizeThatFits:*. Pokud je velikost nového UIViewControlleru, respektive jeho UIView, menší než velikost obrazovky, použije se velikost obrazovky zvýšená o jeden pixel. Tím se zajistí tzv. „bounce efekt“[4]. Poté se již nastaví nová velikost obsahu UIScrollView a velikost schránky.

Nalezení majáku podle lokalizační služby

Jednou z metod nalezení majáku je pomocí lokalizační služby. Pokud uživatel chce nalézt maják dle jeho lokace, musí stisknout ikonu lokalizační služby (viz Obr. 5.7a) a zobrazí se mu nové okno (viz Obr. 5.8), které nalezne majáky v blízkosti uživatele.



Obrázek 5.8: Nalezení majáku dle lokalizační služby.

K tomu byl využit objekt MKMapView, který poskytuje mapové roz-

[4]Obsah, který se vejde do UIScrollView, není již posunutelný. Bounce efekt umožňuje posouvat i nad tímto obsahem. Ve finále to potom vypadá, jakoby obsah skákal.

hraní aplikacím. MKMapView je nastaven na sledování uživatelské polohy. V blízkosti 20 metrů se zaznamená informační maják, který se zobrazí uživateli v UITableView tabulce pod mapou. Všechny majáky, kromě těch s nulovými zeměpisnými souřadnicemi, se zobrazí uživateli na mapě. MKMapView používá protokol MKMapViewDelegate, který v případě změny polohy uživatele notifikuje metodu *mapView:mapView didUpdateUserLocation:*. V této metodě se zkontroluje vzdálenost se všemi informačními majáky. Pokud bude maják v blízkosti 20 metrů, zobrazí se v tabulce a zbarví se do zelené barvy. Majáky ve vzdálenosti větší jak 20 metrů jsou zbarveny červeně a z tabulky smazány. Zobrazeným majákům na mapě se říká anotace a jsou instancí třídy MKAnnotationView. Pro změnu jejich barev je nutné všechny anotace z mapy odebrat a následně znovu přidat. Tato změna se děje při každé změně polohy uživatele. Tím, že se anotace odeberou a znovu přidají, se zavolá metoda *mapView:viewForAnnotation:*, ve které se dle vzdálenosti nastaví barvy majáků.

Hledání majáků dle polohy je povoleno, až když je přesnost GPS vertikálně a horizontálně 10 metrů. Kdyby nebyla uplatněna tato podmínka, při zapnutí GPS kolísá poloha uživatele a nabídne majáky uživateli, i když se u nich zrovna nenachází. To samé platí, když se vypne displej zařízení, na pozadí běží zjišťování současné polohy , které je již přesné a naleze majáky, i když se u nich uživatel právě nenachází. Aby se tomuhle předešlo, je nastaven delegát MKMapView na nil při vypnutí displeje. Při zapnutí displeje zase naopak je delegát MKMapView nastaven pro sledování polohy.

```
- (void)viewDidLoad{
    [[NSNotificationCenter defaultCenter]
     addObserver:self
     selector:@selector(applicationWillResign)
     name:UIApplicationWillResignActiveNotification
     object:NULL];


    [[NSNotificationCenter defaultCenter]
     addObserver:self
     selector:@selector(applicationWillEnterForeground)
     name:UIApplicationWillEnterForegroundNotification
     object:NULL];

}

- (void) applicationWillResign {
    _mapView.delegate = nil;
}
```

```
- (void) applicationWillEnterForeground {
        _mapView.delegate = self;
}
```

Aby se zachytila událost zapnutí a vypnutí displeje, je nutné si zaregistrovat observer na *UIApplicationWillResignActiveNotification* pro detekci vypnutí displeje a *UIApplicationWillEnterForegroundNotification* pro detekci zapnutí displeje.

Operační systém iOS nepodporuje získání informací o okolních WiFi a Bluetooth sítích, které by mohly pomoci určení polohy uvnitř budov. Existují knihovny, které dokáží získat seznam okolních WiFi a Bluetooth vysílačů, ale využívají neokomentované privátní API, které Apple nepovoluje ve svých aplikacích.

Čtení QR kódu a 3-místného klíče

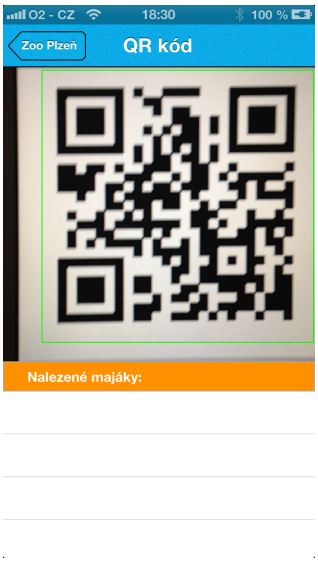
Další metodou nalezení majáku je přečtení QR kódu (viz Obr. 5.9a). K tomu je použita knihovna ZBarSDK (viz 5.3.2). Knihovna ZbarSDK rozkóduje obrázek a unikátní identifikátor majáku se porovná s uloženými majáky, pokud se najde shoda, nabídne se uživateli příslušný maják.

Další metoda nalezení je pomocí 3-místného klíče (viz Obr. 5.9b). K tomu je využito textové pole UITextField. Ten využívá protokol *UITextFieldDelegate* a implementací jeho metody *textChanged:* se zjišťuje velikost zadaného řetězce. Pokud je velikost řetězce rovná třem, porovná se klíč s majáky. Pokud se najde shoda, uživateli se nabídne maják příslušející klíči.

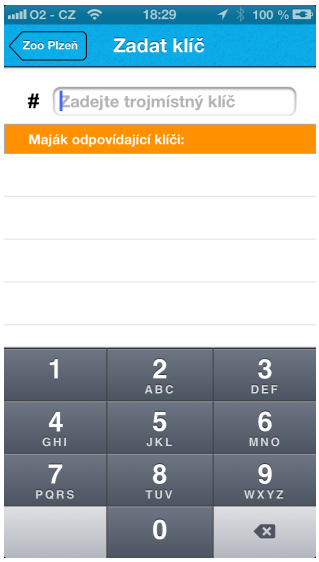
Zobrazení obsahu majáku

Obsah majáku se zobrazí po jeho nalezení. Obsahem může být text, odkaz, obrázky a video z YouTube[5]. Obsah je vytvářen dynamicky, tudíž je možné libovolně kombinovat všechny typy obsahu. Systém Tagiee ale jako první posílá sadu obrázků, pak kombinuje hypertextový odkaz s textem. Text může mít nadpis a odstavec. Nakonec je zobrazeno video z Youtube.

[5]Služba pro sdílení videa.



(a) Nalezení majáku pomocí QR kódu



(b) Nalezení majáku pomocí 3-místného klíče

Obrázek 5.9: Další možnosti nalezení majáků.

Odkaz je řešen pomocí tlačítka UIButton. Text je zobrazen pomocí textového popisku BeaconContentLabel odděděné od UILabel. Od verze iOS 6.0 lze využít k zobrazení textu *NSAttributedString*. NSAttributedString je řetězec, který může být obohacen o různé formátování. BeaconContentLabel přetěžuje metodu *setAttributedText:*, která nastaví potřebné parametry textu.

```
-(void)setAttributedText:(NSAttributedString *)attributedText{
    /* Vytvoreni odstavce */
    NSMutableParagraphStyle *paragraph =
     [[NSMutableParagraphStyle alloc] init];
    /* Zarovnani do bloku */
    paragraph.alignment = NSTextAlignmentJustified;

    /* Atributovany retezec */
    NSMutableAttributedString  *attributtedString =
     [[NSMutableAttributedString alloc]
      initWithAttributedString:attributedText];

    [attributtedString addAttributes:
        @{NSParagraphStyleAttributeName:paragraph}
         range:NSMakeRange(0, [attributtedString length])];

    [super setAttributedText:attributtedString];
    /* Libovolny pocet radek, zalamovani slov */
```

```
    self.numberOfLines = 0;
    self.lineBreakMode = NSLineBreakByWordWrapping;
    /* Prepocitat a nastavit */
    [self sizeToFit];
}
```

Obrázky jsou zobrazeny v horizontálním UIScrollView, ten má pod sebou UIPageControl, který stránkuje jednotlivé obrázky. Po kliknutí na obrázek se otevře galerie MWPhotoBrowser (viz 5.3.4). Obrázky jsou načítány skrze AFNetworking (viz 5.3.1). Již jednou stažený obrázek se uloží do Core Data, tudíž, pokud si uživatel znovu zobrazí obsah, obrázky se načtou z Core Data místo z Internetu.

Youtube video je zobrazeno jako webové okno v UIWebView následovně:

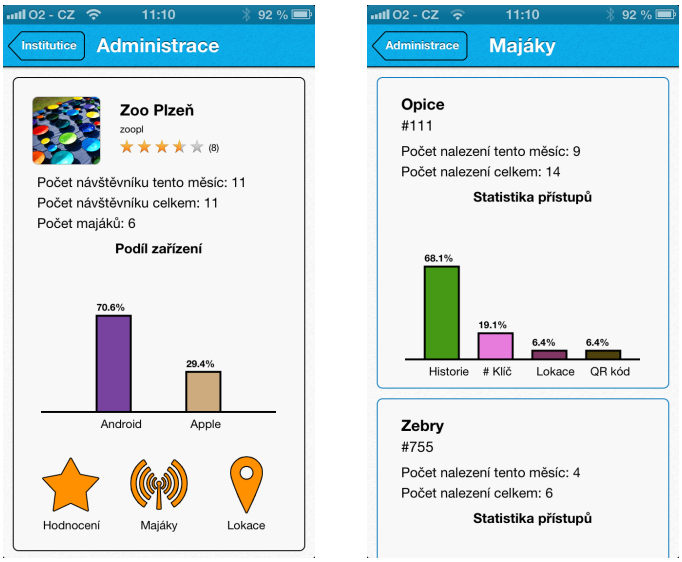
```
NSString *urlString = @"http://www.youtube.com/embed/ID";
NSString *html = [NSString stringWithFormat:@"<html><head>
        <style type='text/css'>body {background-color:
     transparent;color: white;}</style></head>
     <body style='margin:0'><iframe width='%f' height='%f'
     src='%@' frameborder='0' allowfullscreen></iframe>
     </body></html>", 300.0, 200.0, urlString];
UIWebView *videoView = [[UIWebView alloc]
        initWithFrame:CGRectMake(x, y, 300, 200)];
[videoView loadHTMLString:html baseURL:nil];
[self.scrollView addSubview:videoView];
```

Do UIWebView je vložena stránka, kde přes tag *<iframe>* je spuštěno embedované video z Youtube.

Administrátorská část

V rámci předmětu KIV/MKZ vznikla administrátorská část aplikace (viz Obr. 5.10). K administrátorské části se lze dostat z úvodní obrazovky tlačítkem v navigačním menu. Administrátorská část umožňuje získat přehled o institucích, majících, hodnocení uživatelů a umožňuje přidání současné polohy do systému Tagiee. Současná poloha je použitelná ve webovém prostředí systému Tagiee jako poloha majáku.

Přístup do administrátorské části je povolen pouze přihlášeným uživatelům. Přihlašovací údaje jsou stejné jako v případě webového prostředí systému Tagiee. Zaměstnanec instituce má přístup pouze ke svým vytvořeným institucím a majákům.



(a) Seznam institucí. (b) Seznam majáků.

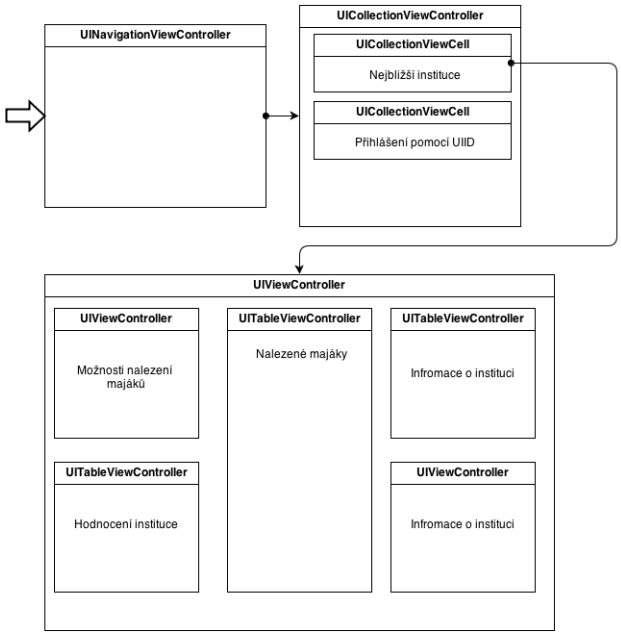
Obrázek 5.10: Administrátorská část aplikace Tagiee.

5.4.4 Verze pro iPad

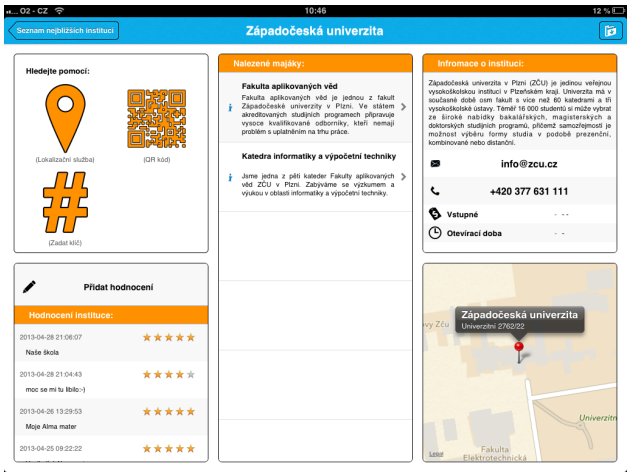
V rámci oborového projektu KIV/OPSWI byla vytvořena verze pro zařízení iPad. Třídy pro práci s databází a komunikaci s Web API jsou společné s verzí pro iPhone. Grafické komponenty jako jsou informace o instituci, obrazovky pro nalezení majáků, seznam nalezených majáků, hodnocení a obsah majáků, byly kompletně převzaty z verze pro iPhone. Uspořádání komponent se změnilo a optimalizovalo pro větší displej tabletu iPad (viz Obr. 5.11).

Základem je UINavigationViewController, seznam nejbližších institucí je zobrazen pomocí UIViewCollectionViewController, který zobrazuje narozdíl od UITableViewControlleru buňky do mřížky. Při přihlášení k instituci jsou na jedné obrazovce (viz Obr. 5.12) zobrazeny informace o instituci, poloha instituce, hodnocení instituce, seznam nalezených majáků a možnosti pro nalezení majáků. Nalezení majáků pomocí GPS, QR kódu a klíče jsou identické s verzí pro iPhone, s tím že místo nového okna je použito modální okno, aby nedocházelo ke zbytečnému překreslování komponent.

Verze pro iPad nebude začleněna v první verzi klientské aplikace do obchodu AppStore, ale vyjde jako aktualizace aplikace.



Obrázek 5.11: Rozložení grafických komponent pro zařízení iPad.



Obrázek 5.12: Obrazovka instituce na zařízení iPad.

5.4.5 Notifikace systému Tagiee

Klientská aplikace notifikuje systém Tagiee o akcích uživatele. Akcemi jsou již zmíněné nalezení majáku uživatelem a přihlášení k instituci. Komunikace probíhá skrze Web API. V případě nedostupnosti připojení se uloží tato notifikace a odešle se až v případě znovu zapnutí aplikace, tedy v metodě *ap-*

plication:didFinishLaunchingWithOptions:. Pro uložení notifikací se využívá Core Data úložiště.

5.5 Publikování aplikace

Klientská aplikace byla odeslána na schválení do digitálního obchodu s aplikacemi App Store. (viz 5.5.1).

5.5.1 App Store

App Store je obchod s digitálním obsahem pro platformu iOS. V lednu 2013 bylo v obchodu dostupných více než 775 tisíc aplikací a to ve 119 zemí světa.

Aby aplikace mohla být zveřejněna v obchodě, musí splňovat podmínky obchodu App Store. Vybrané podmínky týkající se klientské aplikace Tagiee jsou v příloze (viz příloha A.2).

5.5.2 iOS Developer Program

K publikování aplikace do obchodu App Store je nutné si předplatit iOS Developer Program za 99 dolarů ročně. Existuje i iOS Developer Enterprise Program za 299 dolarů ročně, který je určen společnostem a umožňuje kromě klasické distribuce distribuovat aplikace pomocí vlastního obchodu určeného jen pro zaměstnance společnosti.

Pokud si uživatelé koupí aplikaci, Apple si bere provizi 30% a zbylých 70% jde vývojáři. Stejné je to i s transakcemi uvnitř aplikace, Apple si nárokuje také provizi 30%.

5.5.3 iTunes Connect

Po předplacení iOS Developer Programu dostane vývojář přístup do iTunes Connect, který dovolí vývojáři nahrát svou aplikaci do obchodu App Store. Sekce iTunes Connect poskytuje informace o aplikaci, počet jejího stažení,

počet nákupů, informace o platbách a možnost editování popisku aplikace v App Store.

5.5.4 Příprava certifikátů

Pro odeslání aplikace do AppStore je nutné v Member Center pod Identifiers založit App ID. App ID je unikátní řetězec specifikující aplikaci. Klientská aplikace má com.tagiee.Tagiee. Pokud je App ID nastavené, vytvoří se v Provisioning Profiles Distribution certifikát pro odeslání aplikace do iTunes Connect.

5.5.5 Vytvoření aplikace v iTunes Connect

V iTunes Connect se vytvoří aplikace a vyplní se informace v následujícím pořadí:

- Výchozí jazyk aplikace.
- Název aplikace.
- SKU Number - Unikátní identifikátor aplikace.
- Bundle ID - Identifikátor, který umožní budoucí aktualizace aplikace. Bundle ID se bere z vytvořeného App ID.
- Kdy bude dostupná aplikace v App Store.
- Cenová politika.
- Verze aplikace. Musí být 1.0 a vyšší.
- Copyright, kategorie v App Store, věkové omezení.
- Metadata aplikace - Popisek, klíčová slova(podle kterých se bude aplikace hledat v App Store), odkaz na podporu.
- Kontaktní informace na vývojáře.
- Nahrání ikony a ukázkových obrázků z aplikace.

5.5.6 Nahrání aplikace

Pokud je již aplikace hotová (naprogramovaná), už jí zbývá pouze nahrát do iTunes Connect na schválení. V iTunes Connect se zmáčkne tlačítko *Ready to Submit*. V Xcode se vybere z menu Product->Archive. Otevře se Organizer, ve kterém je tlačítko *Distribute...*. Stisknutím tohoto tlačítka se otevře dialogové okno, ve kterém je potřeba vybrat *Submit to the App Store*. Tím se nahraje aplikace k iTunes Connect.

5.5.7 Schvalovací proces

Odeslaná aplikace má svůj schvalovací proces. Schvalovací proces má několik etap. Z těchto etap vyberu jen ty, které se týkají odeslání aplikace skrze Xcode a klientské aplikace Tagiee.

- **Prepare for Upload** - První stav aplikace, čeká se na vyplnění korektních metadat (klíčová slova, popis aplikace, ...) o aplikaci.
- **Waiting for Upload** - Metadata jsou vyplněna, je nutné nahrát aplikaci skrze Xcode.
- **Waiting for Review** - Aplikace je odeslána, ale díky velkému množství odeslaných aplikací čeká ve frontě na schválení. Tato operace trvá obvykle týden až dva.
- **In Review** - Aplikace je ve schvalovacím procesu, zaměstnanci Applu testují aplikaci a provádí na ní automatizované testy. Apple udává statistiku schvalování (viz Obr. 5.13).
- **Pending Developer Release** - Aplikace byla schválena, čeká se na povolení vývojáře, aby umístil svou aplikaci do obchodu.
- **Processing for App Store** - Aplikace byla povolena vývojářem, aplikace se indexuje do App Store. Aplikace by se měla objevit v App Store do 24 hodin.
- **Ready for Sale** - Aplikace byla úspěšně umístěna v App Store.
- **Rejected** - Aplikace nevyhovuje podmínkám obchodu App Store a je vrácena na přepracování. Pracovníci Apple kontaktují vývojáře pomocí emailu nebo telefonicky.

- **Metadata Rejected** - Aplikace byla zamítnuta kvůli špatným metadatům (špatně vyplněný popis aplikace, klíčová slova apod).
- **Removed from Sale** - Aplikace byla vyřazena z obchodu kvůli nově ustanoveným podmínkám.
- **Missing Screenshot** - Obrázky aplikace pro App Store nesplňují podmínky.

Obrázek 5.13: Statistika, kolik odeslaných aplikací na schválení je vyřízeno.

5.5.8 Schválení aplikace Tagiee

V prvním schvalovacím kole nebyla aplikace schválena pro vstup do obchodu App Store z důvodu nepochopení účelu aplikace hodnotící komisí. Funkčnost aplikace je závislá na poloze uživatele. Doporučená poloha pro testování aplikace byla uvedena ve zprávě pro komisi, avšak komise vyžaduje video, které demonstruje funkčnost systému Tagiee. Celá zpráva komise:

We began the review of your app but are not able to continue because we need access to a video that demonstrates your app in use. You can provide a link to a demo video of your app in iTunes Connect. Go to "Manage Your Applications,"select your app, click "Edit Information,"then scroll to the "Review Notes"section and add the demonstration video access details.

Na základě požadavků komise bylo vytvořeno video a odesláno na schválení. V době odevzdání diplomové práce je klientská aplikace ve schvalovacím procesu.

6 Ověření funkcionality

Systém Tagiee byl ověřen na vytvořené instituci Západočeské univerzity v Plzni a její prohlídkou pomocí mobilních aplikací.

6.1 Systém Tagiee

Na Západočeské univerzitě bylo vytvořeno 12 majáků. Každý maják je detekovatelný pomocí QR kódu, unikátního klíče a NFC tagu. Majáky po nalezení zobrazí informace v podobě textů, obrázků, odkazů či videí o daném místě. Seznam majáků je následující:

- **Fakulta aplikovaných věd** - Fakulta aplikovaných věd se nachází v budově na Univerzitní 22 a lze ji najít i pomocí GPS lokace.
- **Katedra tělesné výchovy a sportu** - Katedra tělesné výchovy a sportu se nachází v budově na Univerzitní 14 a lze ji také nalézt pomocí GPS lokace a WiFi sítí.
- **Fakulta elektrotechnická** - Fakulta elektronická se nachází v budově na Univerzitní 26 a též ji lze nalézt pomocí GPS souřadnic.
- **Menza ZČU Bory** - Menza Bory je umístěna na univerzitním trojúhelníku a lze ji také detekovat pomocí GPS polohy.
- **Menza ZČU Kollárova** - Menza Kollárova je na rozdíl od předchozích majáků umístěna v centru města a ne v univerzitním areálu. Lze ji také nalézt pomocí GPS polohy na adrese Kollárova 19 v Plzni.
- **Knihovna Bory** - Knihovna bory je umístěna v univerzitním areálu na adrese Univerzitní 18 a je nalezitelná také pomocí GPS souřadnic.
- **Katedra informatiky a výpočetní techniky** - Katedra informatiky je v budově Fakulty aplikovaných věd ve 4tém patře.
- **Učebna počítačové grafiky** - Učebna počítačové grafiky je ve čtvrtém patře katedry informatiky a výpočetní techniky. Učebna je detekovatelná pomocí WiFi sítí.

- **Učebna počítačových sítí** - Učebna počítačových sítí je ve čtvrtém patře katedry informatiky a výpočetní techniky. Učebna je detekovatelná pomocí WiFi sítí.
- **EEG ERP učebna** - Učebna EEG a ERP je ve čtvrtém patře katedry informatiky a výpočetní techniky. Učebna je detekovatelná pomocí WiFi sítí.
- **Laboratoř digitální fabriky** - Laboratoř digitální fabriky je ve třetím patře katedry průmyslového inženýrství a managmentu. Učebna je detekovatelná pomocí WiFi sítí.
- **Kancelář UK422** - Kancelář Doc. Ing. Přemka Brady MSc. PhD. je ve čtvrtém patře katedry informatiky a výpočetní techniky. Učebna je detekovatelná pomocí WiFi a Bluetooth sítí.

Vytvořená instituce umožní nově příchozím studentům k orientaci v areálu Západočeské univerzity. Návštěvníci získají informace o jednotlivých budovách Univerzity. Při návštěvě menz se uživateli zobrazí odkaz na jídelníček a při návštěvě knihovny odkaz pro vyhledávání knih apod.

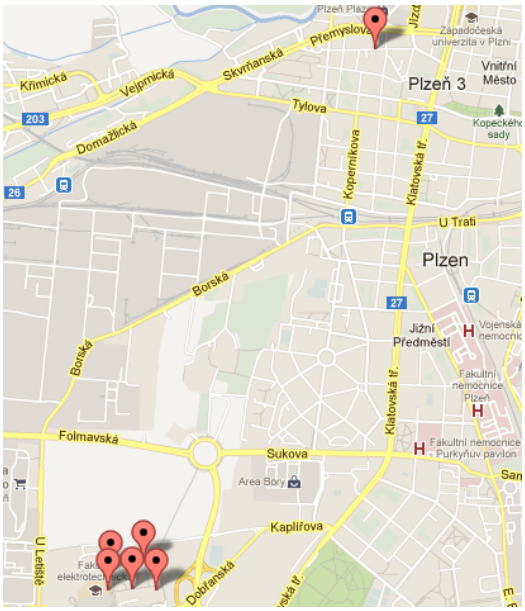
6.1.1 Mapa majáků

Na obrázku (viz Obr. 6.1) je mapa všech majáků, které jsou detekovatelné pomocí GPS souřadnic.

6.2 Webové prostředí

Instituce a majáky byli přidány pomocí webového prostředí na adrese tagiee.com. Webové prostředí bylo testováno na následujících prohlížečích.

- Safari 6.0.3 - OS X 10.8.3
- Chrome 25 - OS X 10.8.3
- Opera 12.15 - OS X 10.8.3
- IE 10 - Windows 7 SP1 x64



Obrázek 6.1: Mapa majáků instituce ZČU nalezitelných pomocí GPS souřadnic.

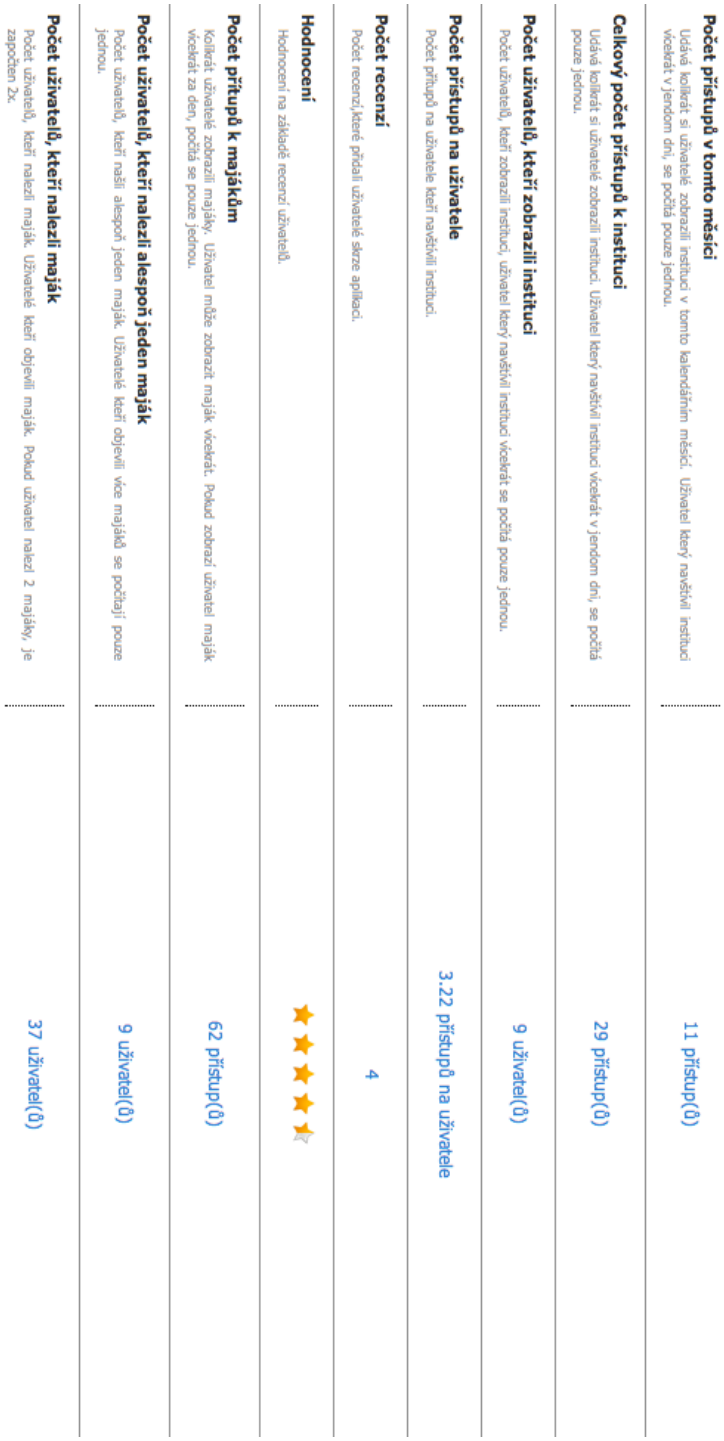
6.2.1 Statistiky

Systém sbírá statistiky o využívanosti instituce Západočeské univerzity (viz Obr. 6.2). Statistiky byly sesbírány za 20 dní existence instituce. Statistiky jsou aktuální k datu 14.května 2013.

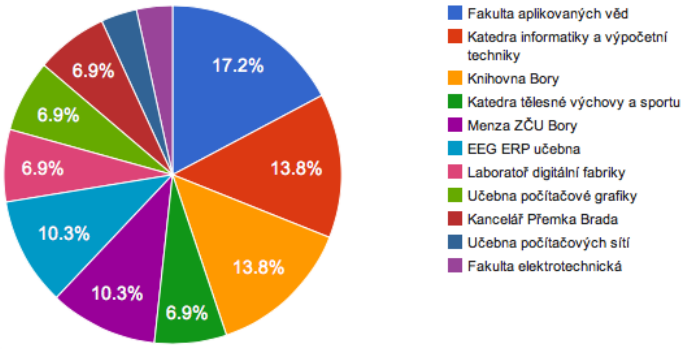
Dalšími statistikami jsou koláčové informace o majácích, tedy procentuální nalezení majáků (viz Obr. 6.3a), podíl technologií, které byly využity pro nalezení majáků (viz Obr. 6.3b) a podíl mobilních platforem (viz Obr. 6.3c).

6.3 Klientská aplikace Tagiee

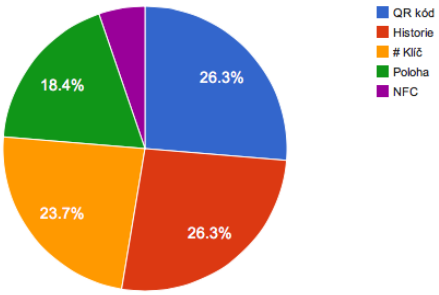
Mobilní aplikace Tagiee byla ověřena na vytvořené instituci Západočeské univerzity (viz 6.1). Postupně byly nalezeny všechny majáky, všemi detekovatelnými možnostmi, které systém iOS nabízí, tedy pomocí zeměpisné polohy, klíče a QR kódu. Zbylé technologie NFC, Wifi a Bluetooth byly vyzkoušeny na platformě Android v diplomové práci „Komunikace a identifikace informačních majáků“ vypracované Bc. Jiří Hromádkou.



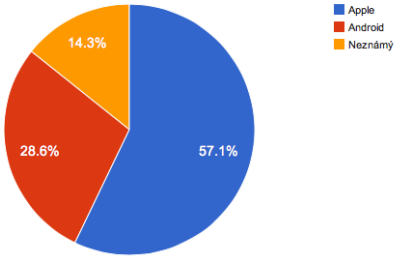
Obrázek 6.2: Statistiky o využití Západočeské univerzity skrze systém Tagiee.



(a) Statistika majáků.



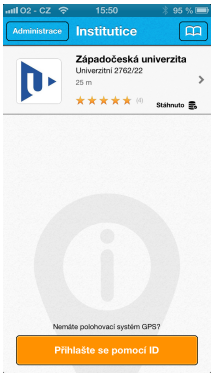
(b) Statistiky o využití technologií pro detekování majáků.



(c) Podíl platforem.

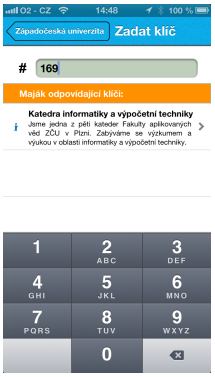
Obrázek 6.3: Koláčové statistiky.

Klientská aplikace byla spuštěna a otestována na zařízení iPhone 4, 4S, 5 a iPad 2. Testování proběhlo i bez přístupu k internetu a bez podpory lokalizačních služeb. V případě vypnutí lokalizační služby se dá přihlásit k Západočeské univerzitě pomocí unikátního identifikátoru „zcu“. Při nedostupnosti internetového připojení se ukládá aktivita uživatele a při příštím zapnutí aplikace se uložená aktivita odešle systému Tagiee. Během prohlídky Západočeské univerzity v Plzni, byly pořízeny obrázky pro ověření funkčnosti (viz Obr. 6.4).

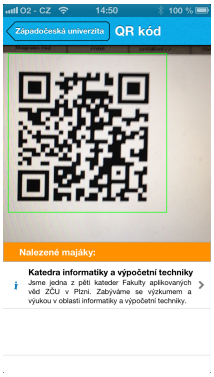


(a) Nalezená Západočeská instituce.

(b) Nalezení majáku pomocí GPS.



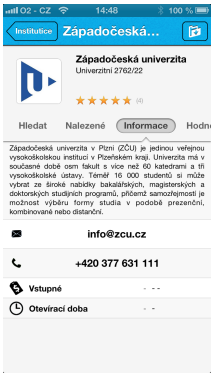
(c) Nalezení majáku pomocí klíče.



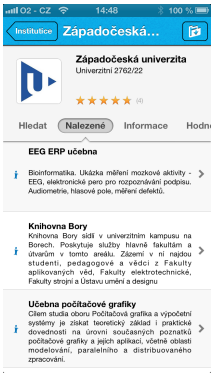
(d) Nalezení majáku QR kódem.

(e) Maják katedry informatiky a výpočetní techniky.

(f) Maják fakulty aplikovaných věd.



(g) Informace o instituci.



(h) Seznam již nalezených majáků.

Obrázek 6.4: Ověření klientské aplikace Tagiie.

7 Závěr

Systémy pro poskytování informací lze využít téměř ve všech oblastech a odvětvích. Tyto systémy umožní uživatelům získat právě ty informace, které potřebují. Díky rozšířenosti byla vybrána mobilní zařízení jako poskytovatelé těchto informací.

Výsledkem je univerzální systém pro poskytování informací Tagiee. Systém umožňuje skrze webové rozhraní zaměstnancům institucí přidat svou instituci do systému spolu s informačními majáky, které uživatelům či návštěvníkům poskytnou informace o instituci. Návštěvník bude k informacím přistupovat pomocí mobilní aplikace. Systém Tagiee poskytuje Web API, pomocí kterého mobilní aplikace komunikují se systémem a získávají potřebná informace. Uživatelé mobilní aplikace mohou nacházet informační majáky po celé instituci v podobě lokalizačních služeb, WiFi, Bluetooth, NFC, QR kódů a 3-místného klíče.

Spolu se systémem byla vytvořena mobilní aplikace Tagiee pro platformu iOS. Aplikace byla vytvořená s ohledem na pravidla a podmínky obchodu App Store, do kterého byla odeslána na schválení. Mobilní aplikace umožňuje nalezení majáků pomocí lokalizačních služeb, QR kódů a 3-místných klíčů. Nalezení pomocí WiFi a Bluetooth není operačním systémem podporováno.

Ověření systému a mobilní aplikace proběhlo na vytvořené instituci Západočeské univerzity v Plzni. Západočeská univerzita má 11 majáků rozmístěných po areálu na Borech a v centru města.

Systém by se v budoucnu dal rozšířit o systém soutěží. Pokud by uživatel nalezl všechny majáky instituce, vyplnil by kontaktní údaje a mohl být zařazen o slosování cen. Nebo by mohla probíhat soutěž o nejlepší fotografii, kterou by návštěvník pořídil v průběhu prohlídky a nahrál ji do systému skrze mobilní aplikaci, kterou by pak zaměstnanci vyhodnotili. Naopak rozšířením aplikace by mohly být notifikace, které by zaměstnanec instituce poslal všem uživatelům, kteří navštívili instituci a informovat je o novinkách či slevách.

Seznam obrázků

Seznam zkratek

- **API**, *Application Programming Interface* - sbírka procedur, funkcí či tříd nějaké knihovny (ale třeba i jiného programu nebo jádra operačního systému), které může programátor využívat. API určuje, jakým způsobem se funkce knihovny volají ze zdrojového kódu.
- **GPS**, *Global Positioning System* - globální družicový polohový systém s jehož pomocí lze určit polohu s přesností na deset metrů.
- **NFC**, *Near Field Communication* - slouží ke komunikaci mezi dvěma zařízeními na krátkou vzdálenost.
- **QR kód** - prostředek pro automatizovaný sběr dat.
- **RSS** - rodina XML formátů určených pro čtení novinek na webových stránkách
- **PMDP**, *Plzeňské městské dopravní podniky, a.s.* - hromadná městská doprava města Plzeň.
- **URL**, *Uniform Resource Locator* - slouží k přesné specifikaci umístění zdrojů na Internetu.
- **WiFi** - označení množiny standardů IEEE 802.11 popisující bezdrátovou komunikaci v počítačové síti.
- **HTTP**, *Hypertext Transfer Protocol* - internetový protokol určený pro výměnu hypertextových dokumentů ve formátu HTML
- **PHP**, *Hypertext Preprocessor* - je skriptovací programovací jazyk.
- **HTML**, *HyperText Markup Language* - je značkovací jazyk pro hypertext. Je hlavním z jazyků pro vytváření stránek v systému World Wide Web, který umožňuje publikaci dokumentů na Internetu.

- **CSS**, *Cascading Style Sheets* - jazyk pro popis způsobu zobrazení stránek napsaných v jazycích HTML, XHTML nebo XML.
- **SŘBD**, *Systém řízení báze dat* - software zajišťující práci s databází.
- **AJAX**, *Asynchronous JavaScript and XML* je obecné označení pro technologie vývoje interaktivních webových aplikací, které mění obsah svých stránek bez nutnosti jejich znovunačítání.
- **MVC**, *Model-View-Controller* rozděluje datovou vrstvu, řídící logiku a uživatelské rozhraní do tří nezávislých komponent.
- **MVP**, *Model-View-Presenter* - MVP je podobná architektuře MVC. Obě architektury se liší hlavně v úloze jejich centrálního kamene, tedy Presenter × Controller. Presenter hraje čistě roli prostředníka, který jen volá model a výsledky předává view, kdežto Controller má navíc na starosti i některé události uživatelského rozhraní.
- **XSS**, *Cross-Site Scripting* je metoda narušení webových stránek zneužívající neošetřených výstupů. Útočník pak dokáže do stránky podstrčit svůj vlastní kód a tím může stránku pozměnit nebo dokonce získat citlivé údaje o návštěvnících.
- **XML**, *Extensible Markup Language* - obecný značkovací jazyk. Jazyk je určen především pro výměnu dat mezi aplikacemi a pro publikování dokumentů, u kterých popisuje strukturu z hlediska věcného obsahu jednotlivých částí, nezabývá se vzhledem.
- **JSON**, *JavaScript Object Notation* - odlehčený formát pro výměnu dat. Je jednoduše čitelný i zapisovatelný člověkem a snadno analyzovatelný i generovatelný strojově.
- **SQL**, *Structured Query Language* - dotazovací jazyk používaný pro práci s daty v relačních databázích.
- **DOM**, *Document Object Model* - je objektově orientovaná reprezentace XML nebo HTML dokumentu
- **UI**, *User Interface* - uživatelské rozhraní (grafické komponenty, okna).
- **UIID**, *Unique Institution ID* - unikátní identifikátor instituce v systému Tagiee.
- **UBID**, *Unique Beacon ID* - unikátní identifikátor majáku v systému Tagiee.

- **UDID**, *Unique Device ID* - unikátní identifikátor zařízení v systému Tagiee.
- **BLOBs**, *Binary Large OBjects* - objemná data například obrázky nebo zvuky.
- **MAC adresa**, *Media Access Control* - jedinečný identifikátor síťového zařízení.

Literatura

[afn()] *AFNetworking* [online]. [cit. 10.5.2013]. Dostupné z: https://github.com/AFNetworking/AFNetworking.

[cha()] *Google Chart Tools* [online]. [cit. 10.5.2013]. Dostupné z: https://developers.google.com/chart/.

[dyr()] *DYRateView* [online]. [cit. 10.5.2013]. Dostupné z: https://github.com/dyang/DYRateView.

[ecg()] *Draw charts with quartz2d for iPhone and iPod* [online]. [cit. 10.5.2013]. Dostupné z: http://code.google.com/p/ecgraph/.

[ego()] *EGOTableViewPullRefresh* [online]. [cit. 10.5.2013]. Dostupné z: https://github.com/enormego/EGOTableViewPullRefresh.

[hus()] *Huston Zoo* [online]. [cit. 20.4.2013]. Dostupné z: http://www.houstonzoo.org/about-the-zoo/.

[ios()] *Develop Apps for iOS* [online]. [cit. 20.4.2013]. Dostupné z: https://developer.apple.com/technologies/ios/.

[ize()] *iZeeum* [online]. [cit. 20.4.2013]. Dostupné z: http://www.izeeum.com.

[jcr()] *JCrop* [online]. [cit. 10.5.2013]. Dostupné z: http://www.jcrop.com.

[jqu()] *jQuery* [online]. [cit. 10.5.2013]. Dostupné z: http://jquery.com.

[jso()] *Introducing JSON* [online]. [cit. 20.4.2013]. Dostupné z: http://www.json.org.

[lun()] *Proč Lunchtime ?* [online]. [cit. 20.4.2013]. Dostupné z: http://o.lunchtime.cz.

[map()] *Google Maps API* [online]. [cit. 10.5.2013]. Dostupné z: https://developers.google.com/maps/.

[mwp()] *MWPhotoBrowser — A simple iOS photo browser* [online]. [cit. 10.5.2013]. Dostupné z: https://github.com/mwaterfall/MWPhotoBrowser.

[mys()] *Using Triggers* [online]. [cit. 20.4.2013]. Dostupné z: http://dev.mysql.com/doc/refman/ 5.0/en/triggers.html.

[net()] *Nette Documentation* [online]. [cit. 20.4.2013]. Dostupné z: http://doc.nette.org.

[ope()] *OpenUDID* [online]. [cit. 10.5.2013]. Dostupné z: https://github.com/ylechelle/OpenUDID.

[ora()] *iOS Version Statistics – February 14th 2013* [online]. [cit. 20.4.2013]. Dostupné z: http://www.14oranges.com/2013/02/ios-version-statistics---february-14th-2013/.

[php()] *PHP (2) - Jak to funguje* [online]. [cit. 20.4.2013]. Dostupné z: http://www.linuxsoft.cz/article.php?id_article=172.

[stu()] *Study finds smartphone users top 1B for first time ever* [online]. [cit. 20.4.2013]. Dostupné z: http://appleinsider.com/articles/12/10/17/study-finds-smartphone-users-top-1b-for-first-time-ever.

[upl()] *Uploadify* [online]. [cit. 10.5.2013]. Dostupné z: http://www.uploadify.com.

[zbd()] *ZBar iPhone SDK* [online]. [cit. 10.5.2013]. Dostupné z: http://zbar.sourceforge.net/iphone/sdkdoc/.

[zoo()] *Zoo Plzeň v číslech* [online]. [cit. 20.4.2013]. Dostupné z: http://www.zooplzen.cz/o-nas/zoo-v-cislech/zoo-v-cislech.aspx.

A Přílohy

A.1 Web API

V této části uvedu plnou dokumentaci k Web API systému Tagiee. Web API má sloužit mobilním aplikacím k získání informací o institucích a majících.

A.1.1 add_review

```
[type] => add_review
[udid] => string - unikatni identifikator zarizeni
[uiid] => string - instituce, ke ktere se hodnoceni pridava
[review] => string - kratke slovni hodnoceni
[rating] => int
[platform] => string {ios|android|win}
[version] => string - ve formatu x.x.x

ODPOVED

[status] => boolean
```

A.1.2 beacon_found

```
[type] => device_found_beacon
[udid] => string - unikatni identifikator zarizeni
[uiid] => string - instituce, ke ktere se pozadavek tyka
[tech] => string {gps|qr|nfc|key} - pouzita technologie
[discovered_datetime] => datetime
[platform] => string {ios|android|win}
[version] => string - ve formatu x.x.x

ODPOVED

[status] => boolean
```

A.1.3 device_visit_institution

```
[type] => device_visit_institution
[udid] => string - unikatni identifikator zarizeni
[uiid] => string - instituce, ke ktere se pozadavek tyka
[discovered_datetime] => datetime
[platform] => string {ios|android|win}
[version] => string - ve formatu x.x.x

ODPOVED
```

```
[status] => boolean
```

A.1.4 get_institution_by_uiid

```
[type] => get_institution_by_uiid
[udid] => string - unikatni identifikator zarizeni
[uiid] => string - instituce, ke ktere se pozadavek tyka
[platform] => string {ios|android|win}
[version] => string - ve formatu x.x.x

ODPOVED

[status] => boolean
[response]=>
(
  [revision] = int
  [logo] => string - url obrazku
  [uiid] => string
  [name] => string
  [description] => string
  [address] => string
  [lat] => double
  [lon] => double
  [tel] => string
  [support_email] => string
  [prices] =>
    (
      [1..N array]
        [type] => string
        [price] => string
        [currency] => string
    )
  [opening_hours] =>
    (
      [1..N array]
        [days] => string
        [time] => string
    )
  [allow_offline] => boolean
  [allow_uploads] => boolean
  [allow_reviews] => boolean
  [number_reviews] => int
  [rating] => float
  [user_posted_review] => boolean
  [beacons] =>
    (
      [1..N array] =>
        (
          [ubid] => string
```

```
          [name] => string
          [description] => string
          [lat] => double
          [lon] => double
          [key] => int
          [additional] =>
            (
              [1..N array] => - sila signalu urcena poradim
                (
                  [tech] => string
                  [data] => string
                )
            )
          [content] =>
            (
              [1..N array] =>
                (
                  [type] => string {image|text|youtube|href}
                  [heading] => string
                  [content] => string
                  [resource_id] =>
                    (
                      [0..N array] => int
                        )
                )
              )
          [discovered] => boolean
        )
    )
  [resources] =>
    (
      [1..N array] =>
        (
          [id] => int
          [type] => string
          [size] => double
          [path] => string
          [metadata] =>
            (
              [width] => int
              [height] => int
              [caption] => string
            )
        )
     )
   )
)
```

A.1.5 get_institution_list

```
[type] => get_institution_list
[udid] => string - unikatni identifikator zarizeni
[platform] => string {ios|android|win}
[version] => string - ve formatu x.x.x
[lat] => double
[lon] => double

ODPOVED

[status] => boolean
[response]=>
(
        [1..N array]
        [revision] = int
        [logo] => string - url obrazku
        [uiid] => string
        [name] => string
        [address] => string
        [lat] => double
        [lon] => double
        [rating] => double
        [number_reviews] => int
)
```

A.1.6 get_institution_revision

```
[type] => get_institution_revision
[udid] => string - unikatni identifikator zarizeni
[uiid] => string - instituce, ke ktere se pozadavek tyka
[platform] => string {ios|android|win}
[version] => string - ve formatu x.x.x

ODPOVED

[status] => boolean
[response]=>
(
        [revision] = int
)
```

A.1.7 get_reviews

```
[type] => get_reviews
[udid] => string - unikatni identifikator zarizeni
[uiid] => string - instituce, ke ktere se pozadavek tyka
[platform] => string {ios|android|win}
[version] => string - ve formatu x.x.x
```

```
[from] => int - spodni limit
[to] => int - horni limit

ODPOVED

[status] => boolean
[response]=>
(
        [1..N array]
        [added] =>  string
        [rating] => double
        [review] => string
)
```

A.2 App Store Review Guidelines

V této sekci budou uvedeny podmínky vztahující se k aplikaci Tagiee pro vstup aplikace do App Store. Podmínky se vztahují ke dnu 5. května 2013. Podmínky jsou pro úplnost uvedeny v originálním anglickém znění. Celé znění App Store Review Guidelines bude uložené na přiloženém CD.

A.2.1 Functionality

- Apps that crash will be rejected
- Apps that exhibit bugs will be rejected
- Apps that include undocumented or hidden features inconsistent with the description of the App will be rejected
- Apps that use non-public APIs will be rejected
- Apps that read or write data outside its designated container area will be rejected
- Apps that download code in any way or form will be rejected
- Apps that install or launch other executable code will be rejected
- iPhone Apps must also run on iPad without modification, at iPhone resolution, and at 2X iPhone 3GS resolution
- Apps that duplicate Apps already in the App Store may be rejected, particularly if there are many of them, such as fart, burp, flashlight, and Kama Sutra Apps.
- Apps that are not very useful, unique, are simply web sites bundled as Apps, or do not provide any lasting entertainment value may be rejected
- Apps that are primarily marketing materials or advertisements will be rejected
- Multitasking Apps may only use background services for their intended purposes: VoIP, audio playback, location, task completion, local notifications, etc.
- Apps that browse the web must use the iOS WebKit framework and WebKit Javascript

- Apps that arbitrarily restrict which users may use the App, such as by location or carrier, may be rejected
- Apps must follow the iOS Data Storage Guidelines or they will be rejected

A.2.2 Metadata (name, descriptions, ratings, rankings, etc)

- Apps or metadata that mentions the name of any other mobile platform will be rejected
- Apps with placeholder text will be rejected
- Apps with descriptions not relevant to the application content and functionality will be rejected
- App names in iTunes Connect and as displayed on a device should be similar, so as not to cause confusion
- Small and large App icons should be similar, so as to not to cause confusion
- Apps with App icons and screenshots that do not adhere to the 4+ age rating will be rejected
- Apps with Category and Genre selections that are not appropriate for the App content will be rejected
- Developers are responsible for assigning appropriate ratings to their Apps. Inappropriate ratings may be changed/deleted by Apple
- Developers are responsible for assigning appropriate keywords for their Apps. Inappropriate keywords may be changed/deleted by Apple
- Developers who attempt to manipulate or cheat the user reviews or chart ranking in the App Store with fake or paid reviews, or any other inappropriate methods will be removed from the iOS Developer Program
- Apps which recommend that users restart their iOS device prior to installation or launch may be rejected
- Apps should have all included URLs fully functional when you submit it for review, such as support and privacy policy URLs

A.2.3 Location

- Apps that do not notify and obtain user consent before collecting, transmitting, or using location data will be rejected
- Apps that use location-based APIs for automatic or autonomous control of vehicles, aircraft, or other devices will be rejected
- Apps that use location-based APIs for dispatch, fleet management, or emergency services will be rejected
- Location data can only be used when directly relevant to the features and services provided by the App to the user or to support approved advertising uses

A.2.4 User interface

- Apps must comply with all terms and conditions explained in the Apple iOS Human Interface Guidelines
- Apps that look similar to Apps bundled on the iPhone, including the App Store, iTunes Store, and iBookstore, will be rejected
- Apps that do not use system provided items, such as buttons and icons, correctly and as described in the Apple iOS Human Interface Guidelines may be rejected
- Apps that create alternate desktop/home screen environments or simulate multi-App widget experiences will be rejected
- Apps that alter the functions of standard switches, such as the Volume Up/Down and Ring/Silent switches, will be rejected
- Apple and our customers place a high value on simple, refined, creative, well thought through interfaces. They take more work but are worth it. Apple sets a high bar. If your user interface is complex or less than very good, it may be rejected

A.2.5 Damage to device

- Apps that encourage users to use an Apple Device in a way that may cause damage to the device will be rejected
- Apps that rapidly drain the device's battery or generate excessive heat will be rejected

A.2.6 Personal attacks

- Any App that is defamatory, offensive, mean-spirited, or likely to place the targeted individual or group in harms way will be rejected
- Professional political satirists and humorists are exempt from the ban on offensive or mean-spirited commentary

A.2.7 Privacy

- Apps cannot transmit data about a user without obtaining the user's prior permission and providing the user with access to information about how and where the data will be used
- Apps that require users to share personal information, such as email address and date of birth, in order to function will be rejected
- Apps that target minors for data collection will be rejected

A.2.8 Legal requirements

- Apps must comply with all legal requirements in any location where they are made available to users. It is the developer's obligation to understand and conform to all local laws
- Apps that contain false, fraudulent or misleading representations or use names or icons similar to other Apps will be rejected
- Apps that solicit, promote, or encourage criminal or clearly reckless behavior will be rejected
- Apps that enable illegal file sharing will be rejected
- Apps that enable anonymous or prank phone calls or SMS/MMS messaging will be rejected
- Developers who create Apps that surreptitiously attempt to discover user passwords or other private user data will be removed from the iOS Developer Program
- Apps which contain DUI checkpoints that are not published by law enforcement agencies, or encourage and enable drunk driving, will be rejected

A.3 Instalační příručka systému Tagiee

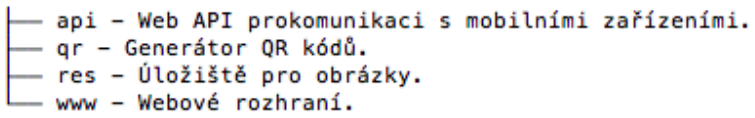
Pro nasazení systému Tagiee na server, je nutné splnit požadavky na systém a dodržet adresářovou strukturu. Poté už stačí překopírovat soubory a systém bude funkční.

A.3.1 Základní požadavky

- Server dostupný na adrese tagiee.com
- MySQL 5.5 a novější s možností vytvoření triggerů.
- PHP 5.3 a novější.
- Schopnost vytvoření poddomén.
- Možnost nastavení práv pro zápis a úpravu souborů.

A.3.2 Adresářová struktura

Pro nasazení je také nutné dodržet tuto adresářovou strukturu :



A.4 Uživatelská příručka webového rozhraní

Na úvodní straně webového rozhraní http://tagiee.com lze vytvořit účet do systému Tagiee, požadavkem na nového uživatele je zadání přihlašovacích údajů a informace o uživateli (bydliště a kontaktní údaje). Po úspěšné registraci je možno vytvořit novou instituci, kde je vyžadován její název, unikátní identifikátor a umístění. K vytvoření instituce a její editace slouží postranní levé menu. Vytvořená instituce není aktivní a pro její aktivaci je nutné vyplnit tyto údaje:

- Popis instituce.
- Logo instituce.
- Telefonní kontakt na instituci.
- Emailový kontakt na instituci.
- Vyplnit ceník.
- Vyplnit otevírací doby instituce.
- Přidat alespoň jeden maják.

Pro přidání majáku je potřeba jeho název. Nový maják též není aktivní a pro aktivaci se musí vyplnit následující údaje:

- Popisek majáku.
- Přidání obsahu majáku.

Text a odkaz k majáku se přidají přetažením do středového kontejneru. Obrázkový obsah se nepřetahuje, ale po kliknutí na obrázkový element se zobrazí dialogové okno, jenž umožní přidání obrázků. Pokud nejsou nahrány žádné obrázky, rozhraní vyzve k jejich přidání na další stránce. Video pomocí YouTube se přidá obdobně jako obrázek.

A.5 Uživatelská příručka mobilní aplikace

Instalace aplikace se provede jejím stažením z digitálního obchodu App Store, až tam bude uveřejněna. V době psaní diplomové práce ještě nebyla aplikace schválena hodnotící komisí.

Aplikace pro svůj běh vyžaduje připojení k internetu. Po jejím zapnutí se zobrazí výzva k povolení lokalizačních služeb uvnitř aplikace. Pokud jsou lokalizační služby dostupné a uživatel se nachází ve 100 kilometrech od Plzně, nalezne se Západočeská univerzita. V případě nedostupnosti lokalizačních služeb se lze přihlásit k Západočeské univerzitě pomocí jejího identifikátoru „zcu“.

Pro nalezení majáků lze využít GPS. Informační majáky se nacházejí v univerzitním areálu na Borech. V případě nalezení pomocí QR kódu a 3-místného klíče lze využít tyto obrázky a klíče :

(a) Maják katedry informatiky a výpočetní techniky. Lze nalézt i pomocí klíče #169.

(b) Maják fakulty aplikovaných věd. Lze nalézt i pomocí klíče #391.