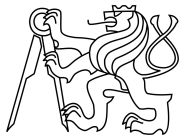
ČESKÉ VYSOKÉ UČENÍ TECHNICKÉ V PRAZE
FAKULTA INFORMAČNÍCH TECHNOLOGIÍ



ZADÁNÍ BAKALÁŘSKÉ PRÁCE

Název:	Tvorba mobilní iOS aplikace pro neziskovou TV
Student:	David Lenský
Vedoucí:	Ing. Tomáš Vondra
Studijní program:	Informatika
Studijní obor:	Softwarové inženýrství
Katedra:	Katedra softwarového inženýrství
Platnost zadání:	Do konce letního semestru 2017/18

Pokyny pro vypracování

CATVUSA je nezisková TV propagující ČR v USA. Nabízí tři televizní kanály (cestopisný, jazykovědný a kuchařský), dvě radiové stanice a mapu zajímavých míst v ČR.

Analyzujte zadané požadavky na mobilní aplikaci a navrhněte jejich řešení na platformě iOS. Aplikace by měla splňovat doporučení pro tvorbu grafického rozhraní iOS a podporovat zařízení iPhone a iPad se systémem iOS 9.0 a vyšším. Aplikaci implementujte. Otestujte funkčnost aplikace a proveďte test použitelnosti nebo beta test s dalšími přispěvateli CATVUSA.

Funkčnost zahrne streaming videa, rádio se zvýhodněním podporovatelů nadace, mapu míst s možností uložení oblíbených a kvízy o ČR. V řešení využijte již navržené REST API, které umožňuje stahování nejnovějších seznamů videí všech kategorií a míst zájmu, a implementujte stranu klienta. Veškerý obsah je již připraven v databázi webu.

Seznam odborné literatury

[1] APPLE INC. iOS Human Interface Guidelines [online]. c2016 [přístup 2016-12-12]. Dostupné z: https://developer.apple.com/ios/human-interface-guidelines/

[2] APPLE INC. App Store Review Guidelines [online]. c2016 [přístup 2016-12-12]. Dostupné z: https://developer.apple.com/app-store/review/guidelines/

[3] SUKHOTIN, Mikhail. Mobile iOS application for a non-profit TV. Praha, 2014. Diplomová práce. České vysoké učení technické v Praze, Fakulta elektrotechnická.

Ing. Michal Valenta, Ph.D.
vedoucí katedry

prof. Ing. Pavel Tvrdík, CSc.
děkan

V Praze dne 12. ledna 2017

České vysoké učení technické v Praze
Fakulta informačních technologií
Katedra softwarového inženýrství

Bakalářská práce

Tvorba mobilní iOS aplikace pro neziskovou TV

David Lenský

Vedoucí práce: Ing. Tomáš Vondra

28. června 2017

Poděkování

Chtěl bych poděkovat Ing. Tomáši Vondrovi za vedení této práce a rady poskytnuté v průběhu její tvorby. Dále bych rád poděkoval Johnu Honnerovi za možnost se k tomuto projektu připojit, za dodání materiálů, zpětnou vazbu k aplikaci a za podporu v její tvorbě. V neposlední řadě bych rád poděkoval Ing. Michaelu Drdlíčkovi za informace o webovém API a rady ohledně problémů, na které bych mohl narazit. Nakonec bych chtěl poděkovat své rodině za neustálou podporu v rámci celého studia.

Prohlášení

Prohlašuji, že jsem předloženou práci vypracoval(a) samostatně a že jsem uvedl(a) veškeré použité informační zdroje v souladu s Metodickým pokynem o etické přípravě vysokoškolských závěrečných prací.

Beru na vědomí, že se na moji práci vztahují práva a povinnosti vyplývající ze zákona č. 121/2000 Sb., autorského zákona, ve znění pozdějších předpisů. V souladu s ust. § 46 odst. 6 tohoto zákona tímto uděluji nevýhradní oprávnění (licenci) k užití této mojí práce, a to včetně všech počítačových programů, jež jsou její součástí či přílohou, a veškeré jejich dokumentace (dále souhrnně jen „Dílo“), a to všem osobám, které si přejí Dílo užít. Tyto osoby jsou oprávněny Dílo užít jakýmkoli způsobem, který nesnižuje hodnotu Díla, a za jakýmkoli účelem (včetně užití k výdělečným účelům). Toto oprávnění je časově, teritoriálně i množstevně neomezené. Každá osoba, která využije výše uvedenou licenci, se však zavazuje udělit ke každému dílu, které vznikne (byť jen zčásti) na základě Díla, úpravou Díla, spojením Díla s jiným dílem, zařazením Díla do díla souborného či zpracováním Díla (včetně překladu), licenci alespoň ve výše uvedeném rozsahu a zároveň zpřístupnit zdrojový kód takového díla alespoň srovnatelným způsobem a ve srovnatelném rozsahu, jako je zpřístupněn zdrojový kód Díla.

V Praze dne 28. června 2017 .....

Odkaz na tuto práci

Lenský, David. *Tvorba mobilní iOS aplikace pro neziskovou TV*. Bakalářská práce. Praha: České vysoké učení technické v Praze, Fakulta informačních technologií, 2017.

Abstrakt

Tato práce se zabývá návrhem a implementací aplikace pro mobilní zařízení se systémem iOS pro projekt CATVUSA. Jedná se o projekt propagující Českou republiku v USA, převážně formou videí o ČR. Účelem práce je zprostředkovat obsah webového portálu projektu uživatelům s prostředím iOS. Práci tvoří analýza podobných projektů, návrh, implementace a testování aplikace.

Klíčová slova iOS, přepracování, implementace, zajímavosti v České republice, Czech-American TV, CATVUSA, streamování videa, mobilní aplikace, Swift

Abstract

This thesis is about designing and implementing an iOS mobile application for CATVUSA project. The goal of this project is to make Czech republic known all over US, using mainly video assets for propagation. The purpose of creating a mobile application is to offer a native way to browse the project content on iOS devices. This thesis consists of similar projects analysis, creating design, application implementation and testing.

Keywords iOS, redesign, implementation, places of interest in Czech republic, Czech-American TV, CATVUSA, video stream, mobile application, Swift

Obsah

Seznam obrázků

Seznam tabulek

Úvod

Czech-American TV (CATVUSA) je nezisková organizace vysílající pořad „John Honner's CATV show“ v šedesáti městech v USA. Jako první a jediná svého druhu vysílá již přes 10 let. Show se snaží americkým divákům přiblížit historii, tradice, kuchyni, architekturu a přírodní krásy České republiky. Zaměřuje se nejen na Česko-Americkou komunitu, ale také diváky se zájmem o české kraje.

Na projektu se podílí mnoho dobrovolníku z České republiky, převážně studenti univerzit v rámci svých bakalářských a magisterských prací. Tím ale jejich spolupráce nekončí. Účastníci v práci často pokračují a přispívají tak k rozšiřování komunity CATVUSA. Na podpoře projektu se dále podílí mnohé české organizace jako například CzechTourism nebo i samotná města, která se ve vysílání objevila.

Hlavní propagační metodou projektu jsou video záznamy, vysílané každý týden v Americké kabelové televizi. Každý díl pořadu je rozdělen do několika částí, ve kterých se představí vždy nějaká tradice, město, památka apod. Video záznam je následně umístěn na internetové stránky, kde jej zájemci mohou zhlédnout. Na tomto webu je možné zhlédnout i další videa, například o českém jazyce či vaření tradičních pokrmů. K dispozici jsou také kvízy o krajích ČR, internetová rádia a nově též česká kuchařka.

Vzhledem k celosvětovému rozvoji mobilních technologií je však potřeba projekt tomuto trendu přizpůsobit. Mobilní zařízení jsou čím dál více chytřejší a výkonnější. V mnoha ohledech jsou tak schopná nahradit dnes již velké a nepraktické počítače. Počet přístupů k internetu z mobilních zařízení každým dnem roste a je tudíž nezbytné informace uživatelům nabízet i v takovéto formě.

Cílem bakalářské práce je pomoci tento prázdný prostor vyplnit. Navrhnout a implementovat aplikaci pro mobilní zařízení se systémem iOS a následně ji poskytnout uživatelům prostřednictvím skrze distribuční platformu AppStore.

Kapitola 1

Cíl práce

Obsahem této bakalářské práce je analyzovat požadavky na aplikaci a navrhnout jejich řešení na platformě iOS. Aplikace by měla být vzhledově i obsahově podobná mobilní aplikaci CATVUSA pro platformu Android. Výstupem práce pak bude hotová aplikace dostupná na AppStore.

Podoba aplikace by měla být založena na vzhledu aplikace pro Android. Z hlediska UX to ale není na platformě iOS úplně nejvhodnější. Uživatelé bez zkušeností s Android aplikacemi by z takového uživatelského rozhraní mohli být zmateni. Z toho důvodu budou převzaty pouze základní prvky a principy aplikace. Vzhled se pak doladí podle doporučených pokynů od Applu.

Aplikace bude realizována pro zařízení iPhone a iPad se systémem iOS 9.0 a vyšším. Hlavním rozdílem mezi verzemi pro telefony a tablety bude UI. Dle [6] a [7] budou verze upraveny, aby splňovaly doporučení na vzhled a funkcionalitu. Pro přehlednost bude verze pro iPhone podporovat pouze orientaci na výšku.

V rámci tvorby verze aplikace pro Android vznikly následující požadavky na produkt.

1.1 Požadavky na mobilní aplikaci

1.1.1 Streamování videa

Základem aplikace je možnost přehrávání videí dostupných na webové stránce. Seznam videí, spolu se všemi potřebnými informacemi o nich, je dostupný přes webové API popsané v sekci 2.3.1. Videa se dělí do tří kategorií (broadcast, class, cooking), podle kterých může uživatel videa filtrovat.

Každé video bude mít stránku s detailem, na které bude uživatel moci video spustit. Dále zde také uvidí doprovodný textový popis celého záznamu. S některými videi je svázán i znalostní kvíz, který se spustí po zhlédnutí videa.

1.1.2 Streamování rádia

Další funkcionalitou aplikace je poslech rádia. K dispozici jsou stejně jako na webových stránkách 2 stanice (Folk a Classic). Uživatel může tyto stanice poslouchat neomezeně až 10 minut. Následně pak bude vyzván k přihlášení do aplikace, popřípadě registraci na webových stránkách, kde bude vyzván k zaplacení předplatného.

1.1.3 Zajímavá místa

Dalším požadavkem je sekce se zajímavými místy v České republice. Budou to například památky, přírodní úkazy, rozhledny či turistická a informační centra. Ty budou zobrazeny ve dvou formách. Ve formě seznamu, ve kterém bude možnost vyhledávat dle názvu, a ve formě interaktivní mapy.

Z obou zobrazení si uživatel bude moci rozkliknout detail daného místa. Zde budou přehledně zobrazeny všechny dostupné informace o tomto místě.

1.1.4 Vědomostní kvíz

V aplikaci budou dostupné vědomostní kvízy, které budou testovat uživatelovu znalost o daném tématu. Tématem kvízu je vždy jeden kraj České republiky. Kvíz bude uživateli automaticky spuštěn po přehrání videa, které se daného kraje týká. Pokud uživatel bude chtít, může si kvíz spustit sám v sekci „Quizzes“. V této sekci najde seznam všech dostupných kvízů.

Každý kvíz obsahuje několik otázek. Uživatel bude předem upozorněn, kolik otázek bude následovat. U každé otázky dostane uživatel na výběr z několika odpovědí, z nichž je vždy pouze jedna správně. Při výběru odpovědi bude uživateli odhalena správná odpověď a možnost pokračovat k další otázce. Na konci kvízu bude závěrečné vyhodnocení s počtem správných a špatných odpovědí. Kvíz si může uživatel kdykoliv zopakovat.

1.1.5 Členství

Uživatel bude mít možnost se v aplikaci přihlásit, pokud se již dříve zaregistroval na webu. Pokud ne, bude mu umožněn rychlý přístup na webovou stránku, kde tak může učinit.

1.1.6 Notifikace

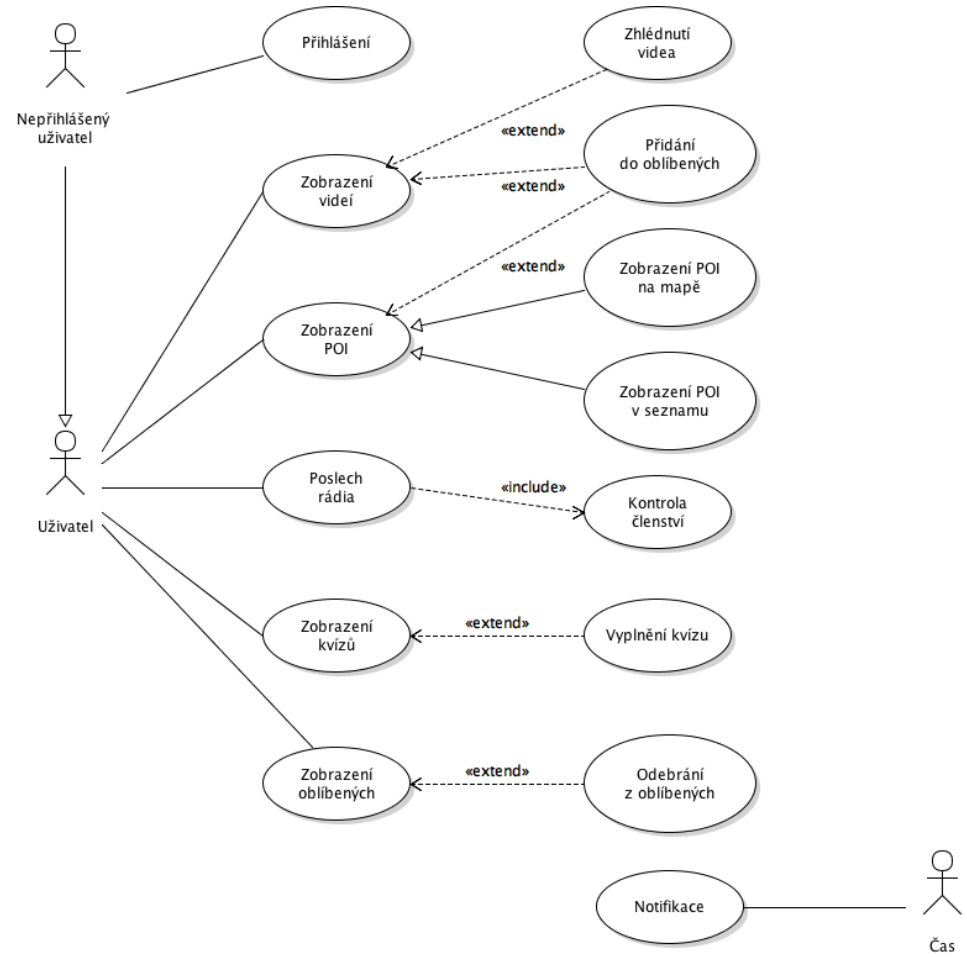
Aplikace bude nabízet možnost automatické kontroly dostupnosti nových videí na webu. Pokud si uživatel tuto funkci zapne a udělí aplikaci oprávnění pro zasílání notifikací, aplikace ho pak upozorní vždy, když se na webové stránce objeví nový záznam.

1.1.7 Sociální sítě

Aplikace bude umožňovat přístup ke stránkám projektu na různých sociálních sítích.

1.1.8 Offline

Aplikace bude schopná fungovat bez připojení k internetu. Ne všechny funkcionality však budou dostupné. Ve video sekci bude seznam videí, která byla dostupná při poslední aktualizaci. Samotná videa však bez internetu nebude možno přehrát. Sekce radio bude pro offline zablokována. Mapy a kvízy budou dostupné, stejně jako seznam videí, s obsahem z poslední aktualizace.



Obrázek 1.1: Use case diagram

Kapitola 2

Analýza a návrh

První část této kapitoly bude věnována anylýze jiných projektů, které mají podobnou tématiku jako CATVUSA. Zaměření bude na projekty s iOS aplikací zabývající se turistikou a naučnými videi o ČR.

Druhá sekce se bude zabývat podrobným rozborem požadavků na aplikaci, jaká řešení má daná problematika a jak bude řešena v této bakalářské práci.

V další části bude návrh datového modelu. V něm bude popsáno webové API, jaká data se z něj čerpají a jak je bude iOS aplikace zpracovávat a ukládat.

Poslední sekce bude zaměřena na návrh uživatelkého rozhraní. Jak bude celá aplikace vypadat, čím je tento návrh podložen a jestli to odpovídá [6].

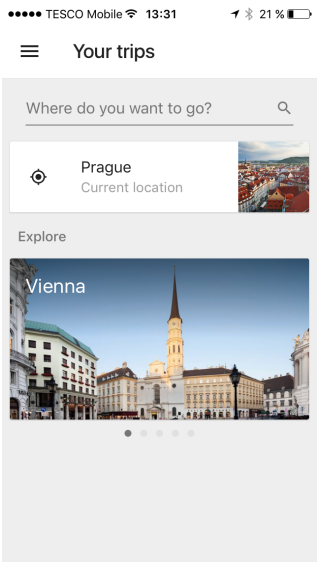
2.1 Analýza existujících projektů

2.1.1 Google Trips

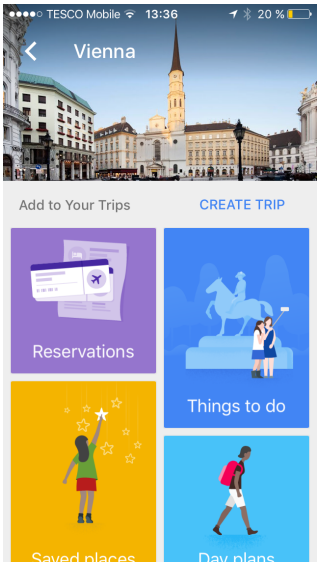
Asi neznámější z následujících aplikací je právě tato [8], od společnosti Google. Fakt, že Google vlastní síť služeb jakou jsou Maps, Calendar nebo Google+, tvoří pro aplikaci Trips skvělý informační základ. Aplikace je tak propojena s ostatními službami a využívá jejich možností. Po vzhledové stránce je aplikace ve stylu Android guidelines (Material design [9]), stejně jako všechny ostatní aplikace Google. Rozdíl mezi Android a iOS verzí je vidět jen na některých ikonkách.

Na hlavní stránce (obrázek 2.1) je vidět vyhledávací pole, návrh jednoho významného místa v okolí (pokud jsme povolili využívání aktuální polohy) a následně návrh významnějších měst v okolních státech. Celou stránku pak doplňuje pro Google specifické „Hamburger menu“, pod kterým najdeme informace o našem Google účtu a základní nastavení.

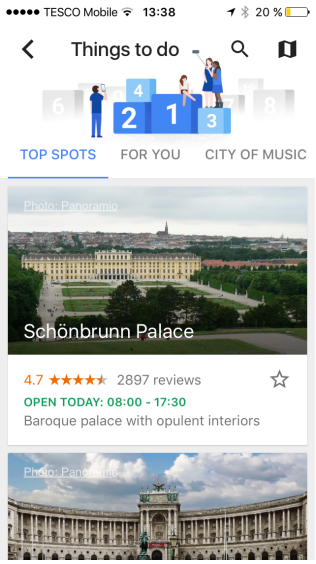
Při rozkliknutí některého z nabízených míst se uživatel dostane na detail (obrázek 2.2). Ten je již o poznání zajímavější. Jsou zde k dispozici



Obrázek 2.1: Přehled



Obrázek 2.2: Detail



Obrázek 2.3: Místa v okolí

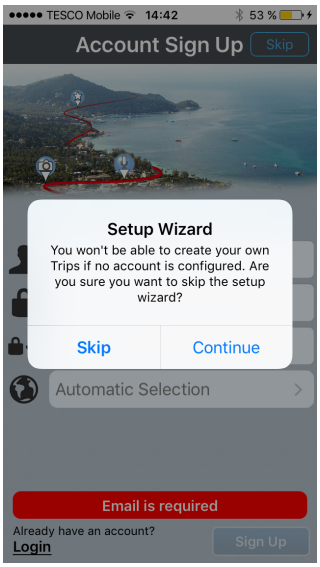
možnosti jako vytvoření vlastní cesty rezervace, uložená místa, jídlo a sekce zábavy („Things to do“). Vytvoření vlastní cesty, přidá aktuální lokaci na hlavní stránku, doplnit uživatel může také datum a další lokace přidružené k cestě. Tím si postupně naplánuje celou dovolenou den po dni. Vytvořenou cestu si pak uživatel může stáhnout a prohlížet offline. Další zajímavou funkcí na detailu je sekce zábavy. Zde uživatel najde přehledně rozdělené aktivity od sportů a venkovních aktivit až po muzea a památky.

Aplikace využívá hodnocení lokalit z jiných aplikací jakou jsou Google maps či sociální síť Google+. Hodnocení, stejně jako recenze, jsou pak přehledně zobrazeny u každé lokality. Stejně tak jsou zde zobrazeny kontaktní informace, pokud jsou dostupné, a mapička. Jediné, co v aplikaci chybí, je sdílení a aktuální události v okolí.

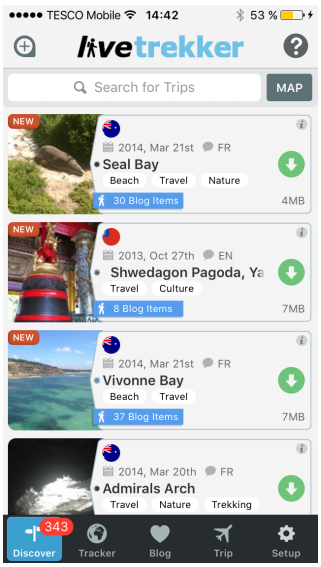
2.1.2 LiveTrekker

Druhá aplikace v pořadí je LiveTrekker [10] od Trekea Mobile, Inc. Ta je spíše takový odstrašující případ. Podklad, jak aplikace nedělat. Aplikace je zmatená, nepřehledná, není jasné jak ji používat a co vlastně umí.

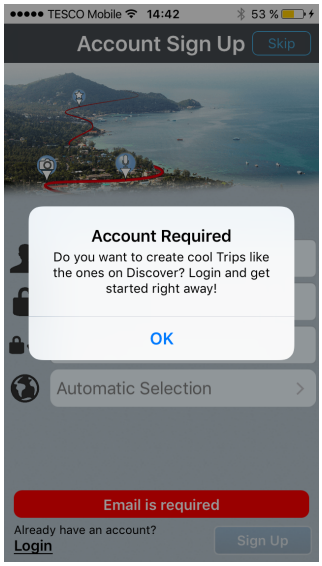
Hned po spuštění na uživatele vyskočí licenční smlouva, kterou musí potvrdit. Po potvrzení se zobrazí stránka s přihlášením. Pro uživatele, kteří účet nemají, je zde tlačítko „Sign up“. Pod tím se pravděpodobně skrývá registrační formulář, nicméně v době testování nebyl funkční. Celý krok přihlášení se naštěstí dá přeskočit pomocí malinkého tlačítka „Skip“ v pravém horním rohu. Po kliknutí vyskočí otravný alert box s otázkou, jestli opravdu chceme



Obrázek 2.4: Přihlášení



Obrázek 2.5: Detail



Obrázek 2.6: Mapa

přeskočit „Setup wizard“.

Po přeskočení přihlášení se uživateli zobrazí hlavní stránka s názvem „Discover“. To je první z pěti přístupných sekcí v TabBaru. Na této stránce vidíme seznam různých tripů z celého světa, seřazených od nejnovějších. Můžeme si zobrazení přepnout i na mapové. Nicméně abychom se podívali co se v daném tripu skrývá, musíme ho stáhnout.

V aplikaci jsou však ještě další 4 sekce. Do těch se ovšem nedá dostat, jelikož aplikace k jejich přístupu vyžaduje přihlášení. Při tom se aplikace přesměruje na přihlášení, kde vyskočí alert box s upozorněním, že je vyžadováno přihlášení. Po jeho odklepnutí musíme znovu opakovat proces s přeskočením přihlášení a tak dále. Ve zkratce je aplikace chaotická a velmi otravná.

2.1.3 Ascape VR: Travel App - 360° World Traveler

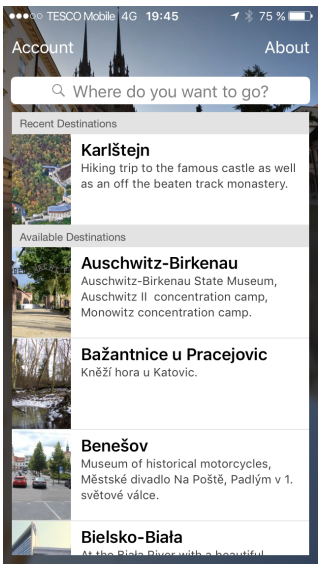
Jako další je aplikace [11] od Ascape. Aplikace nabízí 360stupňové náhledy různých míst po celém světě. Místa si může uživatel prohlížet na mapě nebo v seznamu, popřípadě vyhledávat dle názvu. Virtuální prohlídku si před zhlédnutím musí stáhnout do mobilu. Prohlídky mají dle délky od pár megabytů až po několik stovek megabytů. U každého videa je krátký popis. V době testování zde bylo pouze jedno video z České republiky, a to „Short walk in Prague“.

Aplikace má s touto prací společné pouze téma cestování. Je to však velmi zajímavý nápad. Třeba jako nějaká budoucí funkcionalita?

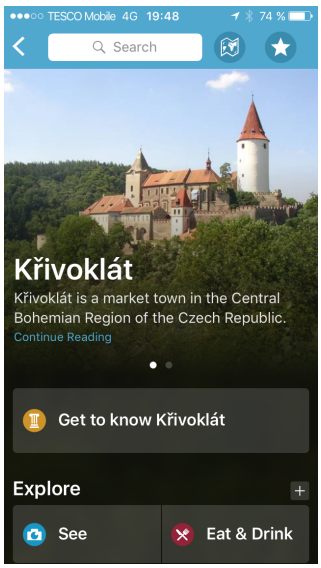
2.1.4 Czech Republic Travel Guide by Triposo

Triposo je cestovní webová stránka a aplikace [12], která vznikla v roce 2011. Od té doby se rozrostla po celém světě. Je možné si stáhnout aplikaci zaměřenou přímo na danou lokaci, kde je veškerý obsah již v základu i offline, nebo paměťově menší aplikaci, do které se jednotlivé lokace stahují jako moduly (pokud je tedy uživatel chce offline). Z hlediska funkcionalit jsou aplikace totožné. Jedině UI je malinko odlišné, nicméně přibližně stejně přehledné.

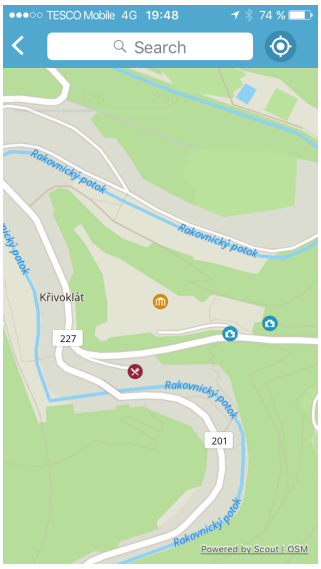
Ačkoliv je aplikace jednoduchá na pochopení, její rozsah je veliký a je zde hodně zanořených stránek. Proto by slovní popis byl příliš matoucí. Bude tedy využito snímků obrazovky jako pomůcky.



Obrázek 2.7: Přehled



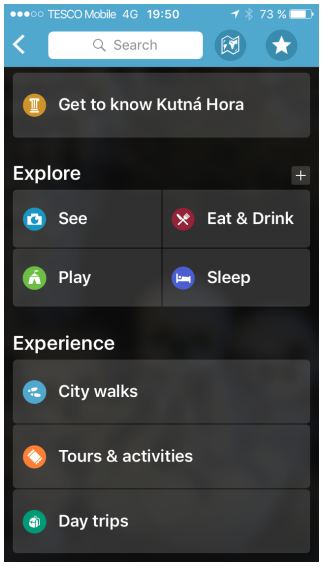
Obrázek 2.8: Detail



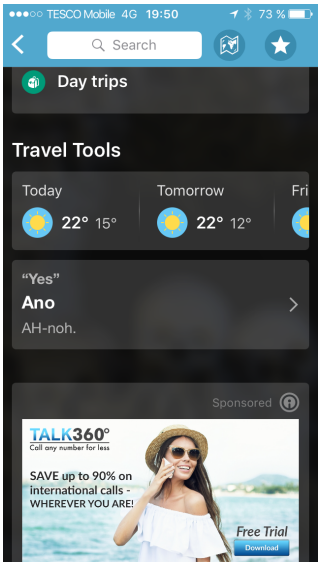
Obrázek 2.9: Mapa

Na obrázku 2.7 je hlavní přehled, kde se uživateli zobrazují nedávno zobrazené destinace a pod nimi všechny ostatní dostupné lokality. Vlevo nahoře je možnost přihlášení, vpravo pak obecné informace o aplikaci a sociální sítě. Pokud uživatel přejde na detail lokace, uvidí obrázek 2.8. Navigace se změní na ikonku zpět, vyhledávací pole, možnost zobrazit lokalitu na mapě (viz obrázek 2.37) a označitelnou ikonku „Oblíbené“. Vyhledávací pole zobrazuje záznamy pouze v rámci destinace. Může tak najít určitý bod zájmu v okolí, nikoliv však v širším měřítku (například celá Česká republika).

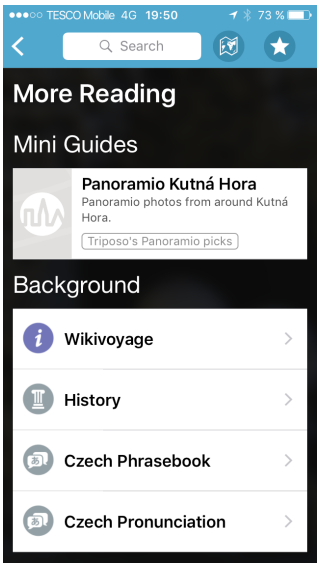
Pod navigační lištou je obrázek lokality a pod ním se nachází mnoho odkazů a nástrojů, jak je vidět na obrázcích 2.10 a 2.11. Úplně na spodu stránky se nachází box s reklamou, která však nikterak nepřekáží. Pokud se jí i přesto chce uživatel zbavit, stačí si zaplatit verzi „Pro“. Na stránce dále můžeme vidět (shora dolu) odkaz na detailní informace u daném místě, kategorie bodů zájmu, nástroje „Experience“, počasí a fráze.



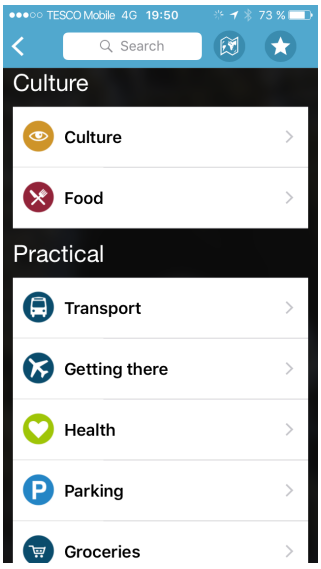
Obrázek 2.10: Detail



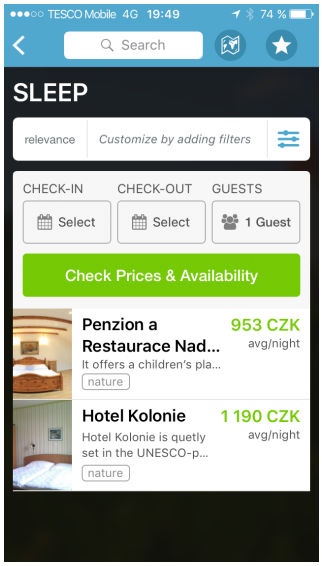
Obrázek 2.11: Detail



Obrázek 2.12: Informace



Obrázek 2.13: Informace



Obrázek 2.14: Booking

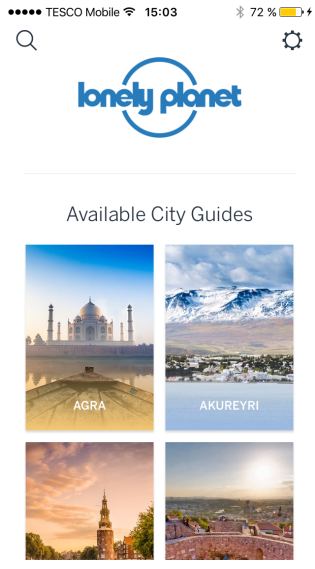
Přes odkaz „Get to know...“ se uživatel dostane na podstránku na obrázcích 2.12 a 2.13. Zde jsou tzv. „Mini Guides“, což jsou malé okruhy s fotodokumentací. Jen některé jsou ovšem dostupné bez „Pro“ verze. Dále zde můžeme vidět odkazy na dodatečné informace o lokaci, jako je například historie, kultura či místní jídlo. Nechybí ani kategorie s praktickými informacemi, jako je doprava, parkování, obchody v okolí a další.

Velmi podstatnou součástí aplikace je také propojení s „Booking.com“, které najdeme v kategorii „Sleep“. Pokoj v hotelu či penzionu je tak možné rezervovat přímo z aplikace. Na stejném principu fungují i nástroje v kategorii „Experience“. Aktivity v této kategorii je možno najít a rezervovat bez nutnosti opustit aplikaci.

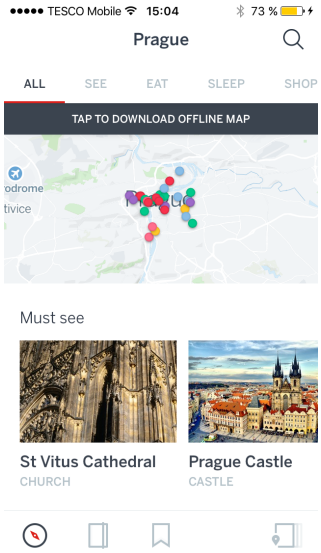
Z aplikací vybraných k analýze v této práci, je právě tato ze všech nejrozsáhlejší z hlediska datového obsahu a funkcí. Zároveň je ale též jedna z nejpřehlednějších a uživatelsky přívětivých. Neplatí zde časté pravidlo „čím více obsahu, tím chaotičtější uživatelské rozhraní“. Aplikace CATVUSA zatím nemá tak rozsáhlý obsah. Do budoucna by však tato aplikace mohla být skvělou inspirací pro změnu UI.

2.1.5 Guides by Lonely Planet

Lonely Planet svou aplikaci [13] vydalo teprve letos (2017) a i přesto má již přes milion stažení. Aplikace slouží spíše jako takový průvodce po větších městech, kterých obsahuje již přes 100. Z České republiky je však zatím dostupná pouze Praha. Při vyhledání slovního spojení se však českých měst zobrazí více, ale místo očekávaného detailu se zobrazí pouze omluvná zpráva, že tohle město zatím není dostupné.



Obrázek 2.15: Hlavní strana



Obrázek 2.16: Přehled

Obrázek 2.17: Detail

Po spuštění se uživatel dostane na hlavní stránku, kde je seznam dostupných lokací (obrázek 2.15). Stránku doplňují pouze dvě ikony, a to klasická lupa pro vyhledávání a ozubené kolečko s nastavením.

Na detailu lokace je uživateli představena swipe navigace, pod kterou se objeví vždy obsah záložky. Záložkami jsou například „All“, „See“, „Eat“, „Sleep“ a další. Pod každou kategorií vidíme interaktivní mapičku s barevnými piny. Každá kategorie má odlišnou barvu. Pod mapou se zobrazí seznam míst a objektů spojených s danou sekcí. Při výběru kategorie „All“ se na mapičce zobrazí všechny piny, barevně odlišené právě podle kategorie, se kterou je svázán.

Každá objekt má vlastní detail s popisem, mapou, kontakty, dopravní dostupností a seznamem zajímavostí v okolí.

Aplikaci nechybí ani systém oblíbených položek. Uživatel si pak může lokace ukládat a plánovat si tak svůj trip. Nechybí ani základní sdílení či ukládání do zařízení. Při trochu detailnějším průzkumu aplikace (ikonek však není mnoho) uživatel narazí i na sekci „Need to know“, kde se dozví mnoho užitečných informací o dané lokaci. Například jak funguje doprava, jaký budget si má připravit, či základní fráze v místním jazyce.

Celkově aplikace působí velmi moderně a přívětivě. Je jednoduchá na používání a uživatel se v ní rychle zorientuje. Má mnoho funkcionalit, a přitom nepůsobí přehlceně. Jedinou nevýhodou je nedostatek lokalit. Z aplikace se však do této práce dá čerpat hodně z hlediska uživatelského rozhraní.

2.1.6 Czech Tourism

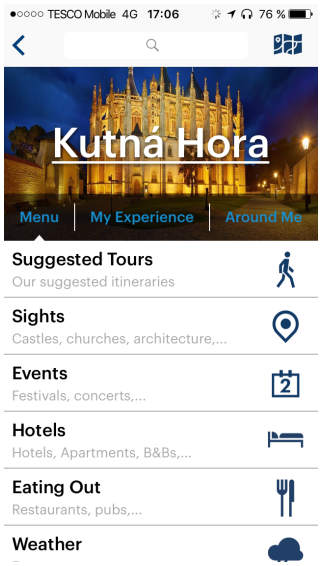
Jako poslední je zde státní agentura CzechTourism, zřízená Ministerstvem pro místní rozvoj ČR. Ta se snaží propagovat Českou republiku jako destinaci cestovního ruchu, jak v zahraničí, tak u nás. To podporuje i fakt, že má CzechTourism na Apple AppStoru 7 aplikací [14], z toho 2 čistě české. Zmíněny zde budou pouze 3 význačnější, které by na tuto práci mohly mít nějaký dopad.

2.1.6.1 Czech Republic – Land of Stories

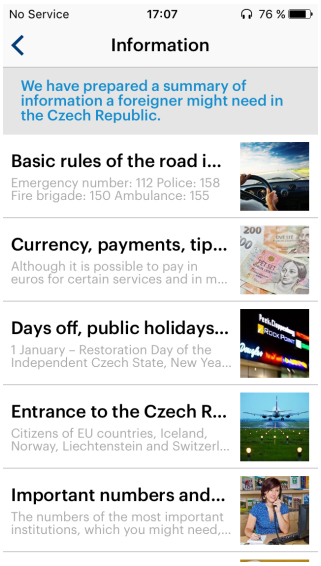
Land of Stories je aplikace zaměřená na vybraná zajímavá města. Přesněji 14. S tímto výběrem se uživatel seznámí hned na hlavní stránce (obrázek 2.18), kde je jejich fotoseznam. Pod informační ikonou (obrázek 2.20) se skrývá mnoho užitečných informací jako jsou ceny všemožných služeb od jídla až po elektřinu, jak funguje zdravotní systém v ČR, doprava, důležitá telefonní čísla, otevírací doby podstatných institucí a další. Co ale uživatele na hlavní stránce zaujme po prvním spuštění, je nastavení. Na ikonce totiž výrazně svítí iOS uživatelům dobře známý „badge“ s číslem. Ten značí, že je dostupná aktualizace, což může být velmi matoucí. Proč by si uživatel měl stahovat aktualizaci něčeho, co zrovna nainstaloval? Aplikace by měla být co nejaktuálnější. Jakýkoliv dodatečný obsah by si měla stahovat, až když ho potřebuje, ne to uživateli nabízet jako dobrovolnou aktualizaci, u které netuší, co se v ní skrývá.

To by k hlavní stránce bylo vše. Detail města (obrázek 2.19) je již mnohem členitější. Najdeme zde vyhledávací pole, ikonku mapy, fotku města,

Obrázek 2.18: Přehled



Obrázek 2.19: Detail



Obrázek 2.20: Informace

navigaci s třemi sekcemi a pod ní obsah dané sekce. Pomocí vyhledávání můžeme najít místa zájmu v oblasti města. Pod ikonkou mapy se skrývá, jak by uživatel očekával, mapka. Co už úplně uživatel neočekává, je fakt, že je mapka vycentrována na jeho pozici a nejbližší okolí a ne na město, o které se zajímá. Dále jsou zde zmíněné tři sekce, které v angličtině nesou názvy „Menu“, „My Experience“, „Around Me“.

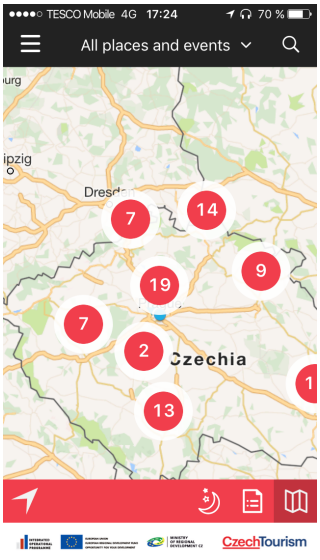
V první najedeme seznam kategorií zajímavých informací, jako jsou restaurace v okolí, počasí, a tak. Dvě, které zaujmou, jsou události a převodník měn. Události jsou něco, v čem má CzechTourism výhodu vzhledem k českému základu aplikace. CzechTourism má mnohem lepší přístup k databázím s událostmi v ČR než zahraniční společnosti. Převodník měn je kapitola sama pro sebe. Je to sice šikovný nástroj, ale vzhledem k umístění budí dojem, že bude převod měny u každého města jiný. Takováto obecná pomůcka by měla být umístěna pod informační ikonkou na hlavní stránce.

Pod záložkou „My experience“ jsou funkce jako oblíbené, itinerář, diář, fotodiář či sdílení na sociální sítě. Pod poslední záložkou „Around me“ se pak skrývá seznam všech bodů zájmu, řazený od nejbližšího.

Aplikace celkově sice není uživatelsky nejpřívětivější či graficky oslnivá, nabízí však velkou výhodu rozsáhlé databáze bodů zájmu či událostí. Bohužel je zde pouze pár dostupných měst.

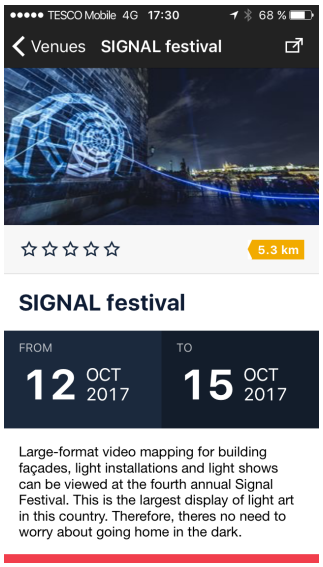
2.1.6.2 Cool Czech Guide

Cool Czech Guide je aplikace velmi jednoduchá, co se týče designu i funkcí. Jejím účelem je poskytnout rychlý přehled bodů zájmu a událostí v okolí.



Obrázek 2.21: Mapa

Obrázek 2.22: Noční



Obrázek 2.23: Detail

Při spuštění se uživateli zobrazí interaktivní mapa České republiky s navigací a toolbarem. V navigační liště najdeme již známé „hamburger menu“, ikonku vyhledávání a filtrační „drop down menu“. V liště nástrojů jsou pak možnosti jako vycentrovat mapu na uživatele, přepnout mezi denními a nočními událostmi a přepnout na mapové či seznamové zobrazení. Tuto obrazovku poněkud narušuje reklama na CzechTourism a další partnery.

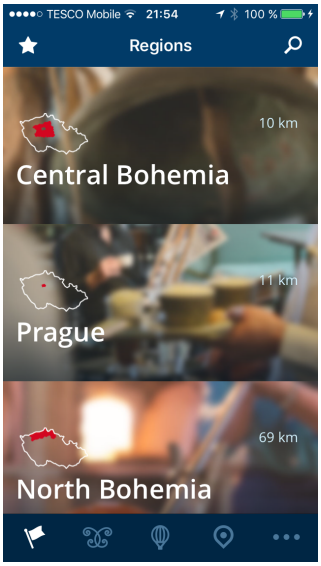
Ve zmíněném „hamburger menu“ se skrývá již známá sekce s praktickými informacemi (jako ve většině aplikací CzechTourism). Dále zde najdeme i přihlášení a další dvě funkce, a to „Favourites“ a „Schedule“. Ty jsou ale podmíněny právě přihlášením.

Svou funkci aplikace plní dobře. Uživatelské rozhraní není nijak překombinované a uživatel se s ním rychle sžije. Jediné, co se aplikaci dá vytknout, je reklama ve spodní části a ne zrovna rozsáhlá databáze událostí a bodů zájmu na to, odkud aplikace pochází.

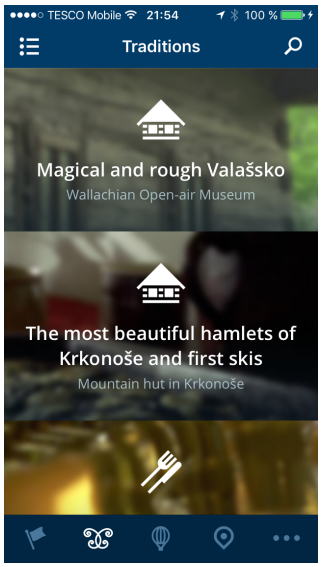
2.1.6.3 Czech Traditions

Na závěr tu je aplikace Czech Traditions, která je obsahově asi nejpodobnější CATVUSA. Její účel je podobný účelu aplikace CATVUSA. Má propagovat Českou republiku pomocí článků a videí o tradicích a významných místech. Videí však není mnoho a primární zaměření je na články. Video a audio záznamy slouží spíše jako dodatečné pomůcky k článkům. Tím se projekt liší. Předchozí projekty sloužily většinou jako průvodci po České republice. Tato aplikace je však zaměřená hlavně na českou kulturu.

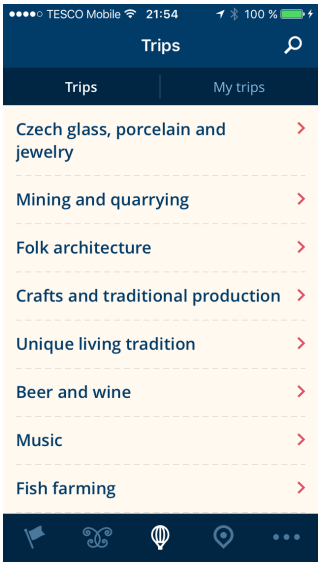
Po spuštění aplikace se dostaneme na stránku se seznamem regionů



Obrázek 2.24: Regiony



Obrázek 2.25: Tradice



Obrázek 2.26: Výlety

České republiky (obrázek 2.24). I přes velké množství obsahu a různých sekcí byla zvolena tabbarová navigace. Tím se aplikace vyhnula nepřehlednému „hamburger menu“. Dále je zde možné přepnout na jiný typ zobrazení, případně vyhledávat výraz v rámci sekce. Při výběru regionu se dostaneme na detail (obrázek 2.29). Ten je ve formě článku o daném regionu. Dále pak také obsahuje video a audio záznamy (pokud k danému tématu existují) a zobrazení dané lokace na mapě (obrázek 2.28).

Sekce tradice (obrázek 2.25) funguje na stejném principu jako předchozí sekce regiony. Jediným rozdílem je, že se článek týká různých řemesel, tradicí, architektury či třeba receptů.

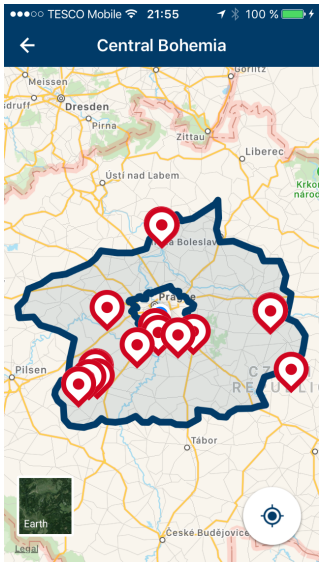
Další sekcí jsou výlety (obrázek 2.26). Zde si může uživatel vybrat z několika předpřipravených výletů, rozdělených do sekcí podle kategorie (hudba, pivo a víno a další). Dále je pak možný vytvořit výlet vlastní, do kterého je možné přidávat dostupné body zájmu z celé republiky.

Poslední zajímavou sekcí je „All in one place“ (obrázek 2.27). Zde se nachází seznam všech bodů zájmu v republice. Uživatel se může podívat na jejich detail a zobrazit je na mapě či je přidat do vlastního výletu.

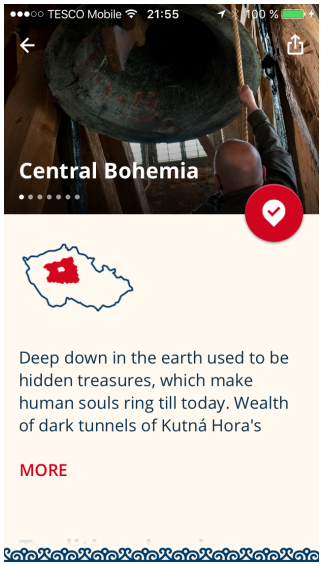
Dále můžeme v tabbaru najít ještě ikonu s třemi tečkami. Pod tou se skrývá jednoduché nastavení s třemi volbami. Přihlášení, video tutoriál k aplikaci a informace o aplikaci.

Aplikace má elegantně řešenou navigaci pomocí tabbaru. Ten je na rozdíl od „hamburger menu“ přehlednější. Všechny sekce jsou vidět přímo a uživatel se proto může jednodušeji seznámit s celým obsahem aplikace. Detaily v jednotlivých sekcích jsou konzistentní a liší se jen v drobnostech. Uživatel

Obrázek 2.27: Overview



Obrázek 2.28: Tradice



Obrázek 2.29: Detail

si tak nemusíš pamatovat několik různých layoutů, což aplikaci dodává na ovladatelnosti. Celkově je uživatelská zkušenost s aplikací velmi dobrá. Aplikaci se dají vytknout pouze dva body. Prvním je rozsah databáze. Na to, že je aplikace z produkce společnosti CzechTourism, je zde velmi málo bodů zájmu. Druhým je to, že v době testování aplikace se nepřehrávala žádná videa.

2.1.7 Závěrem

Aplikace **Google Trips** je sice velmi dobře provedena, nicméně koncipována převážně jako průvodce pro turisty. Proto se její provedení do projektu CATVUSA úplně nehodí. Pokud by však byl v projektu zájem rozvíjet dále sekci „Map“, určitě by stálo za prozkoumání možnosti propojení se službami Google.

Z aplikace **LiveTrekker** se nedá vzít mnoho. Spíše naopak. Je zde mnoho věcí, kterých se vyvarovat. Hlavně však zbytečného omezování uživatele. Proč ho otravovat při spuštění aplikace přihlášením, když není nutné pro běh aplikace? Proč mu přihlášení nutit pokaždé, když se pokusí přistoupit k obsahu, které přihlášení vyžaduje? Uživatel by měl být volbu. Je-li přihlášení pro aplikaci důležité, ale ne nezbytné, mělo by být viditelně dostupné, ne však házeno uživateli pod nohy, při každém kroku.

Triposo se ostatním projektům vymyká jednou zásadní vlastností. A to je absence navigace, která by rozdělovala aplikaci do několika sekcí. Aplikace tak při spuštění nestaví uživatele před desítky možností, ale dává mu jasnou cestu kudy pokračovat. Na detailu již je mnoho nástrojů a možností, ale jsou přehledně rozdělené do sekcí a výstižně textově popsané. Uživatel tak nemusí

klikat na nic neříkající ikony bez popisku ve snaze zjistit, co znamenají. Tento přístup však nemůže být v aplikaci CATVUSA využit vzhledem k několika různým sekcím, které spolu nijak nesouvisí. Dalším kladem Triposo je propojení se službami třetích stran jako je Booking. To by mohlo být pro CATVUSA zajímavé při rozšiřování sekce zájmových bodů, která by se mohla rozvinout do průvodce v podobném stylu jako je Triposo.

Aplikace od **Lonely Planet** klade velký důraz na design a user experience. Aplikace působí velmi čistě, plynule a jednoduše. Uživatel není zahlcen množstvím voleb či funkcí. Cenou za tuto přívětivost je však absence mnoha funkcí jako je jakékoliv plánování, obecné informace a další, které můžeme najít v robustnějších aplikacích. V projektu CATVUSA je však příliš mnoho sekcí pro takovýto přístup. Inspirovat by se však dalo ve specializovaných podstránkách jednotlivých sekcí, které by měly být jednoduché a neměly by zahlcovat množstvím voleb a ikon. Posledním přínosem aplikace by mohla být mapa. Jednoduché rozdělení typu míst zájmu dle barev je velmi přehledné, obzvláště při filtrování. Malé barevné tečky zabírají mnohem méně místa než složitější piny a mapě to přidává na přehlednosti.

Land of Stories od CzechTourism je aplikace velmi omezená obsahově. Je zaměřena na přesný počet měst a proto je velmi konkretizována. Výlety, hotely, restaurace a další místa je tak snažší spravovat. Mimo to má aplikace i pár zajímavých funkcí jako je sdílení na sociální sítě a fotodiář. To by mohlo být využito při rozšiřování aplikace CATVUSA o průvodce. Dále je zde pak velmi podrobná sekce informací o české republice. Ta se do tohoto typu aplikací hodí. Pro uživatele pouze se zajímající o ČR je to poutavé info a pro turisty je to velmi užitečné. Co je však na aplikaci odrazující je design. UI je sice jednoduché, ikony přehledné, ale celkovým dojmem uživatele nenadchne. Aplikace se designem drží velmi striktně základních stavebních prvků od Apple. Prvky nejsou nijak upraveny, není zde nic originálního.

Dalším dílem z produkce CzechTourism je **Cool Czech Guide**. Vzhledem k jednoduchosti aplikace se z ní dá pojmout jen pár věcí. První je rozhodně mapa. Spojování bodů zájmu do hromadného bodu s počtem obsažených pinů je rozhodně dobrým krokem. Obzvláště v aplikaci CATVUSA je to vhodné. Ta sice zatím obsahuje pouze souřadnice všech informačních center v republice, ale jen těch je již dostatek na přehlcení mapy. Spojování bodů zájmu tak rozhodně bude využito. Další dobrou praktikou je změna barvy navigace vzhledem ke stavu aplikace. Uživatel tak má mnohem výraznější optickou zpětnou vazbu o tom, kde se v aplikaci právě nachází. Věcí, které se naopak vyvarovat je reklama na spodu obrazovky. Mapu to výrazně zmenšuje a znepřehledňuje, nemluvě o tom, že to není opticky vůbec přívětivé.

Poslední aplikací je **Czech Traditions**. Prvním dobrým aspektem je navigace. Ta je vždy viditelná a informuje uživatele, kde se právě nachází. Ikony by mohly být doprovázeny ještě textem, jelikož samy o sobě o svém obsahu mnoho nevypovídají. Tak to bude použito v aplikaci CATVUSA. Ta obsahuje několik různých sekcí, které spolu nesouvisí a nejsou provázané. Proto

zde tento typ navigace dává smysl. Druhým použitelným prvkem je horní navigace v sekci „Trips“. Je zde využit SegmentedControl pro přepínání mezi veřejnými a vlastními výlety. To se v CATVUSA bude hodit například na přepínání mezi kategoriemi videí. Dalším povedeným prvkem je mapa. Vyznačení oblasti a pinů je výrazné a přehledné. Označování oblastí by se mohlo využít při rozšiřování mapové části aplikace v budoucnu.

Aplikace **CATVUSA** bude navržena s ohledem na výsledky této analýzy. Budou v rozumné míře využity kladné aspekty ostatních aplikací a bude kladen důraz na uživatelskou přívětivost. Ikdyž aplikace není originální či úchvatná, uživatelé ji stále budou používat. Aplikace uživatele však v žádném případě nesmí žádným způsobem odradit. Ať už je to zmatené / nepřehledné rozložení prvků nebo často vyskakující upozornění, takových chyb je třeba se vyvarovat.

2.2 Analýza požadavků

V této sekci bude provedena analýza požadavků na aplikaci CATVUSA. Požadavky byly vytvořeny v rámci diplomové práce [15].

2.2.1 Video přehrávač

Prvním požadavkem na aplikace je streamování video záznamů z webového portálu CATVUSA. Existuje mnoho způsobů, jak zaimplementovat tuto funkcionalitu do iOS aplikace. Běžnou praxí je však napsat si vlastní rozhraní přehrávače založené na přehrávačích nativních nebo je použít v jejich původní podobě. Další možnosti implementace budou rozebrány dále od těch nejjednodušších po složitější.

První a zároveň nejsnazší na implementaci je využití jednoho z nativních přehrávačů v systému iOS. Ty existují dva, a to MPMoviePlayer [16] a AVPlayer [17]. MPMoviePlayer je jednodušší, má všeobecně známé uživatelské rozhraní a pokrývá všechny potřeby pro základní video playback. Hodí se pro malé projekty, které vyžadují jednoduché přehrávání videa. Co je méně známé, je již fakt, že je tento přehrávač postaven na již zmíněném AVPlayeru. I to je jedním z důvodů, proč je od iOS 9.0 MPMoviePlayer označen jako „deprecated“. V nových projektech se proto doporučuje použít o něco náročnější AVPlayer.

Druhým a momentálně jediným aktuálním nativním přehrávačem je právě zmíněný AVPlayer. Ten je oproti MPMoviePlayeru o něco složitější, avšak nabízí mnohem více funkcionalit. Nevýhodou je absence nativního tlačítka s funkcí „Airplay“, která umožňuje streamovat obraz do ostatních zařízení Apple v síti. Pokud nejsou požadavky na přehrávač náročné a obejdou se bez této funkcionality, obsahuje AVPlayer vlastní jednoduché rozhraní, které je možno použít.

Tabulka 2.1: Video format [5]

Video	Resolution	Audio	File format
H.264 - High Profile Level 4.2	up to 4K, 30fps	AAC-LC, up to 160Kbps, 48kHz	.m4v, .mp4, .mov
MPEG-4 - Simple Profile	up to 2.5Mbps, 640x480px, 30fps	AAC-LC, up to 160Kbps, 48kHz	.m4v, .mp4, .mov
M-JPEG	up to 35Mbps, 1280x720px, 30fps	ulaw, PCM stereo audio	.avi

Pokud jsou však nároky na přehrávač vyšší, je nutno si k postavit jakousi nadstavbu, vrstvu uživatelského rozhraní, k AVPlayeru či MPMoviePlayeru. Vzhledem ke komplexnosti AVPlayeru je však jednodušší se poohlédnout po nějakém frameworku, který již tuto vrstvu obsahuje. Těch je velmi mnoho od jednodušších, neplacených až po plně upravitelné a placené. Ty základní jsou postaveny ještě na starším MPMoviePlayeru, nicméně vzhledem k zastaralosti této knihovny není doporučené je používat. Další kategorií jsou frameworky postavené na AVPlayeru. Bývají mnohem propracovanější a umožňují jednoduché upravení svého vzhledu. To je však stále v rámci jakýchsi zažitých pravidel pro tvorbu UI. Je-li potřeba přehrávač nějakým způsobem neobvyklý, je nutno si UI pro video přehrávač vytvořit vlastními silami.

Všechna předchozí řešení však mají stále jednu velkou nevýhodu a tou je závislost na nativních přehrávačích od Apple. Ty mají velmi omezenou podporu přehrávaných formátů. Pokud je je jedním z požadavků přehrávání videa či audia v jiném formátu než v tabulce 2.1, je potřeba přejít k poslední kategorii přehrávačů. To jsou přehrávače postavené na nástrojích, jako jsou ffmpeg [18] či VideoToolbox [19]. Jedná se o nízkoúrovňové API určené pro kódování a dekódování videa. Výhodou ffmpegu je lepší dokumentace, zaostává však ve výkonu a spotřebě energie. VideoToolbox je nativní API využívané i AVPlayerem. Má tak přístup k hardwaru [20] a lépe s ním spolupracuje. Je proto energeticky méně náročnější. Nicméně není řádně zdokumentovaný, a pro práci s ním je proto nutné mít dostatek zkušeností jak s iOSem vývojem, tak s mediálními technologiemi a formáty. Není však nutné si přehrávač programovat. Existuje již mnoho hotových řešení. Avšak přehrávače využívající tyto nástroje jsou velmi jednoduché nebo placené.

Pro účely této práce bude využita varianta první, a to nativní přehrávač AVPlayer. Požadavek na přehrávač je jediný, a to pouze streamování videa ve formátu MP4 (kodek H.264). Na uživatelské rozhraní nároky nejsou. Pro takovéto zadání AVPlayer plně vyhovuje.

2.2.2 Rádio přehrávač

Druhým požadavkem na aplikaci je streamování audia z webu. Jedná se o dvě „rádio stanice“. Ty reprezentují dva různé žánry skladeb. Skladby nejsou pod DRM ochranou. Přistupuje se přímo k souborům skladeb, není využitá žádná technologie pro streamování mediálního obsahu.

K přehrávání bude stejně jako u videa využit AVPlayer, který je naprosto dostačující. Ovládací prvky budou využity vlastní, jelikož jich nebude mnoho a budou velmi jednoduché. Bude tak umožněna větší kontrola nad přehráváním.

Jedním z požadavků na přehrávání je náhodný výběr skladeb. Skladby jsou však z webového API poskytovány ve formě statického seznamu. Proto bude k náhodnosti a eliminaci častého opakování skladeb využit systém vlastní.

Dalším a posledním požadavkem na rádio přehrávač je kontrola uživatelského členství. Uživatelé, kteří se dosud do aplikace nepřihlásili, budou po 10ti minutách přehrávání vyzváni k přihlášení, popřípadě jim bude umožněno přejít na webové stránky aplikace, kde se budou moci registrovat.

2.2.3 Mapa

Nároky na mapu v aplikaci nejsou nijak neobvyklé. Je potřeba zobrazit dostupná místa zájmu na mapě. Ta by měla být přístupná i ve formě seznamu. Aplikace by měla umožnit vyhledávání v těchto místech a označování za oblíbená (a následně filtrování).

API zatím poskytuje pouze seznam informačních center v České republice. Další body zájmu zatím dostupné nejsou. V budoucnu jsou plány na přidání dalších míst, jako jsou památky, muzea, přírodní úkazy a jiná zajímavá místa. S přibývajícími místy vzniknou i další požadavky na mapu. Například třeba navigace k danému místu či turistické okruhy.

2.2.4 Členství

Projekt CATVUSA umožňuje uživatelům přispět finančně k rozvoji projektu a to rovnou několika způsoby. Prvním je darování libovolné částky projektu. Uživatel tím nezíská žádné výhody, pouze přispěje k prodloužení života projektu. Aplikace by měla uživateli pohodlně tuto možnost zprostředkovat. Jelikož si ale Apple ponechává 30 % z jakékoliv platby v rámci aplikace, bude toto provedeno přesměrováním do prohlížeče Safari, kde se uživateli načte stránka z webu projektu určená právě k příspěvkům. To by sice mohlo být provedeno přímo v aplikaci pomocí UIWebView, nicméně na uživatele nepůsobí placení v „neznámém“ prohlížečí úplně bezpečně. Je proto lepší je přesměrovat do jejich primárního prohlížeče, kde platbu mohou v klidu dokončit.

Druhým typem příspěvku je zakoupení členství. Členství již stojí fixní částku a uživatel z něj získá různé výhody. Těmi jsou například neomezený poslech rádia či přístup ke všem videím. Výhod je však více a budou přibývat.

Členství si uživatel může zakoupit pouze přes webový portál pomocí PayPalu. V aplikaci se pak přihlásí přes zadaný e-mail a heslo. Stejně jako u dobrovolných příspěvků bude aplikace uživatele přesměrovávat do Safari, pokud narazí na nějakou funkci vyžadující členství.

2.2.5 Notifikace

Řešení kontroly nového obsahu je na iOS trochu složitější. Zatímco aplikace na Android si mohou relativně bez omezení spouštět bloky kódu na pozadí, na iOS to možné není. Aplikace musí mít dobrý důvod a musí si zažádat o povolení této funkce. Variant běhu aplikace na pozadí je dokonce několik. To však není jediné řešení. Vzhledem k aktuálnímu stavu serverového API (více v sekci 2.3.1), by ani nebylo nejlepším řešením, aby si na pozadí aplikace žádala o data.

Druhým možným řešením jsou push notifikace. K těm je možné využít nástroj OneSignal. Ten však také potřebuje zásah do serverové strany aplikace. Tento přístup bude preferován, implementace ve finální aplikaci však bude záviset na možnostech programátorů starajících se o web.

2.2.6 Offline

Většina obsahu aplikace je závislá na dostupných datech z webového portálu, nicméně zablokovat přístup do celé aplikace, pokud uživatel bude bez připojení k internetu není z hlediska uživatelské zkušenosti dobré řešení. Vzhledem k nízké frekvenci změn dustupných informací by však aplikaci bylo možné používat i v režimu offline. Stačí si ukládat odpovědi serveru ve formátu JSON na disk, popřípadě cachovat náhledy obrázků. Cachování bude implementováno vlastní, bez použití frameworků. Obrázky se budou ukládat do složky „Cache“ ve souborovém systému aplikace. Tato složka je určená právě k ukládání obsahu, který není nijak zásadní pro běh aplikace. Systém si v případě potřeby může tuto složku promazat (například při nedostatku místa na disku).

Prvním problematickým prvkem jsou videa, jejichž velikost je příliš veliká na automatické uchovávání na disku. Nicméně je ale možné přidat tlačítko s funkcí stáhnout. Uživatel si tak bude moct vybrat, která videa chce stáhnout a uchovávat offline.

Další problém je poslech rádia, který obsahuje velké množství audio záznamů. Ten bude z toho důvodu nutně vázán na připojení k internetu. Stejně na něm bude závislé přihlášení do členství a veškeré odkazy na internetové stránky (například sociální sítě).

Zbylé sekce budou bez problému dostupné offline s daty z doby, kdy byl uživatel posledně připojen k internetu (a používal aplikaci).

2.3 Návrh datového modelu

V této sekci bude analyzováno již navržené a funkční webové API, které je použito ke komunikaci mezi serverem a mobilními aplikacemi. Následovat bude popis datového modelu, znázorňující reprezentaci dat v rámci aplikace.

2.3.1 Webové API

K datům na serveru aplikace přistupuje přes webové API navržené v rámci vývoje Android aplikace. Na API spolupracovali jak autor aplikace, tak webmaster projektu. Výběr řešení velmi zjednodušil fakt, že web je tvořen pomocí redakčního systému WordPress. Vzhledem k povaze aplikace však byly potřeba pouze jednoduché dotazy na data. Proto byl zvolen plugin, se kterým již měl webmaster dobré zkušenosti, a to WordPress REST API(Version 2) [21].

Co se týče struktury dat, tento WordPress plugin podporuje pouze JSON. XML se hodí spíše na složitější a robustnější struktury. Pro přenos jednodušších datových typů je JSON naprosto dostačující a obsahově úspornější. Hodí se tedy pro tento projekt více a omezení tudíž není problémem.

V následující sekci bude rozbor vytvořeného RESTového rozhraní. Budou popsány jednotlivé metody, jejich využití a struktura dat v odpovědi.

2.3.1.1 Video

http://www.catvusa.com/api/video?api=1&type=class

Tato metoda slouží k získání seznamu videí na serveru. Bez parametrů se vrátí seznam všech videí s datovou strukturou popsanou níže. Je možné si však od serveru vyžádat pouze množinu videí pomocí parametru „type“ nebo pouze jeden záznam použitím parametru „id“. Datová struktura vypadá následovně:

```
[
  {
    "id": Int,
    "type": String,
    "name": String,
    "date": Int,
    "url": String,
    "direct_url": String,
    "image": String,
    "content": String,
    "category": String,
    "excerpt": String,
    "chapters": [{
        "desc": String
        "img": {
            "id": Int,
            "alt": String,
            "title": String,
            "caption": String,
            "description": String,
            "mime_type": String,
            "url": String,
            "width": Int,
            "height": Int
            "sizes": {
                "thumbnail": String,
                "thumbnail-width": Int,
                "thumbnail-height": Int,
                "medium": String,
                "medium-width": Int,
                "medium-height": Int,
                "medium_large": String,
                "medium_large-width": Int,
                "medium_large-height": Int,
                "large": String,
                "large-width": Int,
                "large-height": Int
            }
        }
    }]
  },
  {...}
]
```

Většinu parametrů není potřeba vysvětlovat, proto budou dále rozepsány pouze ty nejasné či jinak významné.

- **type** – Nabývá jedné ze tří následujících hodnot: class, cooking, broadcast. Určuje do jaké sekce video patří.
- **url** – Adresa webové stránky, kde se nachází dané video.
- **direct_url** – Adresa video souboru.
- **image** – Adresa náhledového obrázku k videu.
- **content** – Popis obsahu videa.
- **category** – Kategorie videa (např. „Alphabet“ nebo „Czech Regions“).
- **excerpt** – Zkrácený popis obsahu videa.
- **chapters** – Náhrada popisu videa. Je-li video rozděleno na jednotlivé kapitoly, každá z nich má vlastní popis, popřípadě obrázek.
  - **alt** – Vždy prázdný string.
  - **caption** – Vždy prázdný string.
  - **description** – Vždy prázdný string.
  - **mime_type** – Vždy „image/jpeg“.
  - **url** – URL původního obrázku.
  - **width, height** – Velikost původního obrázku. Velikosti jsou pouze dvě a to 320x180 a 1920x1080.
  - **sizes** – Obsahuje odkazy na 4 velikosti původního obrázku. „Thumbnail“, „medium“, „medium_large“ a „large“. Tyto velikosti jsou konstantní, ale nepřekračují velikost původního obrázku. Pokud je tedy původní obrázek velikost 320x180, jsou velikosti „medium_large“ a „large“ stejných rozměrů.

Z popisu je vidět, že API nebylo navrženo úplně nejlépe. Jsou zde jak duplicitní parametry, tak některé naprosto zbytečné. Přesnější návrh změn bude popsán v sekci 2.3.1.6.

2.3.1.2 Rádio

http://www.catvusa.com/api/radio?api=1

Metoda umožňuje přístup k seznamu písní, které je možné přhrávat v aplikaci v sekci rádio. Velkou nevýhodou této metody je chybějící možnost filtrovat dle parametrů (např. „genre“). Je proto nutné vždy projít kompletní

seznam a potřebná data si vyfiltrovat lokálně. Ukázka datové struktury odpovědi:

```
[
  {
    "description": String,
    "file": String,
    "intro": String,
    "gendre": String,
    "image": String
  },
  {...}
]
```

Popis struktury:

- **description** – Popis písně. Obsahuje název, autora a případný krátký text.
- **file** – URL audio souboru.
- **intro** – URL krátkého audio souboru přehrávaného před samotnou písní.
- **gendre** – Žánr písně. Nabývá hodnot „folk“ a „classic“. V názvu parametru je překlep, název by měl být „genre“.
- **image** – URL obrázku k písni.

Tato struktura je jíž výrazně jednodušší než u videa, nicméně záznamů je pořád velmi mnoho a nemožnost filtrovat velmi omezuje práci s touto metodou. Dále by se také vzhledem k množství dat mohly zkrátit vrácené hodnoty. V této datové struktuře jsou 3 dlouhé URL souborů, které jsou potřeba. Kdyby místo nich existovaly metody v API pro přístup k těmto souborům například pomocí unikátního kódu každé písně, mohl by se místo těchto URL posílat pouze tento kód.

2.3.1.3 Mapa

```
http://www.catvusa.com/data/turist.json
```

Tento příkaz vrátí seznam bodů zájmu. Na metodě je zvláštní, že její adresa je rozdílná od předchozích. Není již v sekci „api“, ale „data“. Navíc má koncovku „json“. Vypadá to spíše jako přímý přístup k souboru bez snahy o přeadresování v rámci RESTového API. Tento nekonzistentí přístup by měl být v příští verzi odstraněn. Vrácená data jsou v následujícím formátu:

```
[
  {
    "id": String(numeric),
    "name": String,
    "city": String,
    "street": String,
    "postal_code": String,
    "latitude": Double,
    "longitude": Double
  },
  {...}
]
```

Tato odpověď nepotřebuje v podstatě žádné vysvětlení. Názvy parametrů vyjadřují jejich význam a hodnoty jsou krátké. Rozdělení adresy do několika komponent je vhodné, jelikož ne vždy je potřeba zobrazit všechny informace. Jediné, co se dá vytknout, je textová (i když obsahuje pouze čísla) podoba identifikátoru, který je v ostatních metodách reprezentován číselně jako integer.

2.3.1.4 Kvíz

```
http://www.catvusa.com/api/quizzes?api=1
```

Pomocí tohoto příkazu je možné získat seznam kvízů ze serveru. Odpověď serveru vypadá takto:

```
[
  {
    "info": {
        "name": String,
        "main": String,
        "results": String,
        "level1": String,
        "level2": String,
        "level3": String,
        "level5": String,
        "level5": String
    },
    "questions": [{
        "a": [{
            "option": String,
        }],
        "q": String,
        "correct": String,
        "incorrect": String,
        "select_any": Bool,
        "force_checkbox": Bool
    }],
    "extra": ???,
  },
  {...}
]
```

Popis struktury:

- **info** – Informace o kvízu. Zbytečné pole, jeho obsah by mohl být na nejvyšší úrovni viz popis prvků v něm.
  - **main** – Vždy prázdný string.
  - **results** – Vždy prázdný string.
  - **levelX** – Vždy string ve formátu „Level X“. Využití neznámé. Parametry jsou zbytečné.
- **questions** – Pole otázek.
  - **a** – Pole odpovědí k otázce. Vždy obsahuje tři prvky. Název parametru by měl být „answers“. Není potřeba název zkracovat.
    - **option** – String s textem odpovědi.
  - **q** – String s textem otázky. Vhodnější název parametru by byl třeba „text“.

  - **correct, incorrect** - String s textem, který má být vypsán při správném / špatném zodpovězení otázky. Jsou to jednoduché texty jako „You're correct!“. Ty jsou v API naprosto nepotřebné. Není to obsah, který by si aplikace nemohla sehnat sama. Navíc není potřeba mít tyto dvě věty u každého záznamu několikrát duplikované.
  - **select_any** – Vždy false.
  - **force_checkbox** – Vždy false.

- **extra** – Vždy NULL.

Kvízů naštěstí není mnoho, jinak by tato odpověď zabírala velké množství dat, stejně jako video. Je zde mnoho parametrů, které se vůbec nepoužívají, jsou nesmyslné nebo bez vysvětlení nedávají smysl. Absence dokumentace se na této sekci promítá nejvíce. Tak, jak je datová struktura vrácena nyní, by se jí dala téměř polovina zahodit a ušetřit tak podstatné množství dat a umožnit rychlejší přenos.

2.3.1.5 Uživatelská sekce

http://www.catvusa.com/wp-json/wp/v2/users/me

Metoda umožňuje ověřit zadané přihlašovací údaje uživatele. V odpovědi se pak vrací zda přihlášení bylo úspěšné či ne. Nevrací se žádný specifický token, pouze potvrzení. Adresa metody se znovu viditelně liší od všech ostatních. Zde to však nemusí být nutně problémem implementačním nýbrž omezení WordPressu a jeho pluginů. Použitý plugin bohužel umí využívat pouze basic authentication.

Při využití této metody je potřeba přidat do hlavičky parametr s názvem „Authorization“ a jako jeho hodnotu poslat řetězec tvořený z uživatelského jména a hesla a dále zakódovaný pomocí kódování Base64.

Tento způsob zabezpečení není vůbec bezpečný, jelikož se vše dá rozkódovat bez potřeby dalších informací. Nicméně v rámci projektu uživatel nevytváří žádná vlastní data ani nemůže vlastní data nikam nahrát. Kdokoliv by se tedy přihlásil přes jeho účet, nezíská žádné osobní informace kromě těch, které už má. A samozřejmě přístup k prémiovému obsahu projektu CATVUSA.

2.3.1.6 API v2.0

Problémy zmíněné v předchozích sekcích by se měly opravit do příští verze API. Spolu s nimi by se měly objevit dvě důležité funkce. První úpravou bude dokumentace. Bez té je API nepřehledné a mnoho parametrů bez dalšího kontextu nedává smysl. Druhou je metoda pro zjištění data poslední úpravy. V aktuální verzi musí aplikace stahovat přibližně 1 MB, pokud chce aktualizovat informace. Ve většině případů aplikace data stahuje zbytečně, jelikož je má

nacachované. Tím, že by se aplikace serveru pouze ptala na datum poslední změny a stáhla nové informace pouze v případě, že přibyl nový obsah, by se zamezilo zbytečnému plýtvání daty.

Stejně tak by pomohlo filtrování obsahu dle parametrů. Klientská strana by mohla API žádat například pouze o jednu kategorii rádia, o určitý druh POI či všeobecně pouze o záznamy přidané od určitého data.

Další výraznou změnou, šetřící data, by bylo nahradit plné cesty odkazů třeba za určitý identifikátor, který by se následně používal ve spolupráci s dalšími metodami v API. Například u videa máme „url“, „direct_url“, „image“ a následně 5 URL obrázků u každé „chapter“. To je dohromady něco mezi 3–30 adresami na video. Přitom jsou si adresy velmi podobné. Navíc videa i obrázky již mají vlastní unikátní ID. Stačilo by tedy vytvořit například metody v API, které by vracely obsah těchto adres.

Datově náročné jsou též duplicitní popisky. Místo jejich zkrácené verze by se mohlo posílat pouze počet znaků, které má popisek obsahovat. Další nepříjemností je, že jsou popisky ve formátu HTML. To by nebyl takový problém, kdyby byly obsahem pouze textové tagy. Popisy však obsahují i obrázky a to velmi negativně ovlivňuje práci s nimi. Dále obsahují menší nekonzistenci, a tím jsou konce řádků, najít se dají jak „\n“, tak tag „
“.

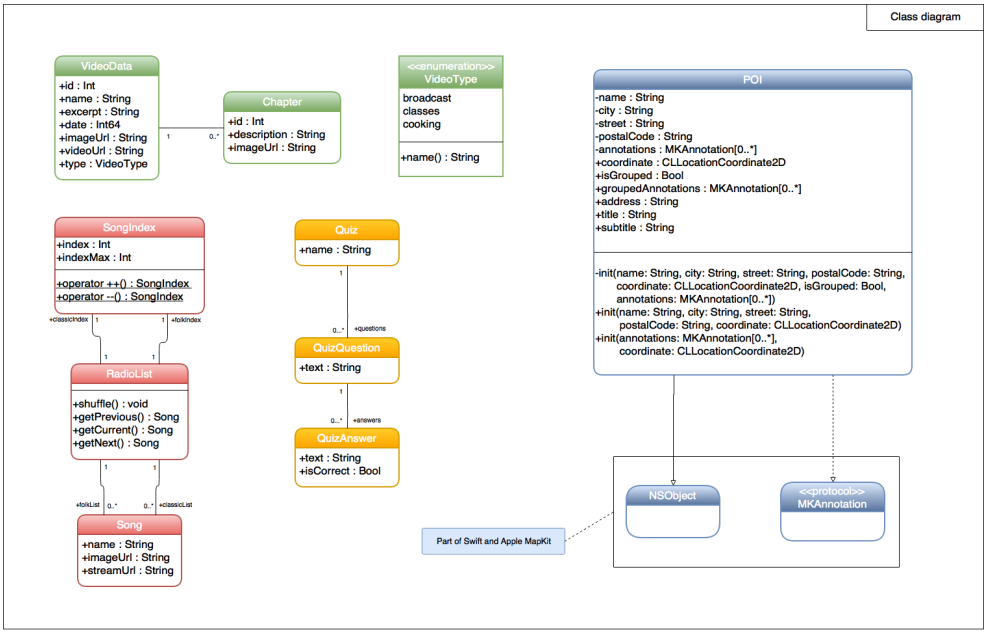
Další problém se týká parametrů. Parametry by měly mít výstižné názvy, tedy ne názvy jako „a“ či „q“. Parametry, které nedávají smysl a jsou prázdné nebo pokaždé stejné, by buď měly mít vysvětlení v dokumentaci, nebo měly být odstraněny.

V neposlední řadě jsou zde URL adresy metod RESTového API. Názvy by měly být sjednoceny a umístěny pod jednotnou cestu.

2.3.2 Datový model

Na obrázku 2.30 je schéma datového modelu, rozdělené do sekcí dle barev.

- **Video (zelená)**
  - **VideoType** – Je enumerace používaná k identifikaci kategorie videa. Metoda *name()* vrací název této kategorie v čitelné podobě.
  - **Chapter** – Je struktura reprezentující část popisu videa. Každé video může mít několik kapitol, z nichž každá má vlastní popis. Ten může obsahovat jak text, tak obrázek.
  - **VideoData** – Reprezentuje veškerá data potřebná k prezentaci videa uživateli. Obsahuje stručný popis videa, název, datum, typ videa, URL náhledu, URL videa a seznam kapitol (Chapters).
- **Rádio (červená)**
  - **Song** – Struktura drží data o jednotlivých písních. Každá píseň má název, URL audio souboru a URL obrázku.



Obrázek 2.30: Class diagram

  - **SongIndex** – Specificky upravená struktura reprezentující index aktuálně přehrávané písně.
  - **RadioList** – Struktura spojující seznamy písní a jejich indexy. Umožňuje navigaci v těchto seznamech a též náhodné přeházení pořadí písní.

- **Kvíz (žlutá)**
  - **Quiz** – Reprezentuje data o jednotlivých kvízech. Je zde název kvízu a seznam otázek.
  - **QuizQuestion** – Struktura obsahující text otázky a seznam možných odpovědí.
  - **QuizAnswer** – Jednotlivé odpovědi na otázky. Ukládá se text a zda je odpověď správná, nebo ne.

- **Mapa (modrá)**
  - **POI** – Třída reprezentující zajímavé místo na mapě. Třída dědí od třídy NSObject a implementuje protokol MKAnnotation. Tím je zajištěna možnost předávat tuto třídu přímo do MKMapView, která daná místa zobrazuje na mapě. Třída má několik privátních atributů. Jsou to části adresy, která se dá získat přes veřejný parametr „address“. Anotace mohou být i sloučené v případě, že jich

je mnoho blízko sebe. Pak jsou jednotlivé anotace míst uloženy v atributu „annotations“. Jsou pak přístupné přes veřejný atribut „groupedAnnotations“. „Title“ a „subtitle“ jsou nutné atributy protokolu MKAnnotation. Jejich obsah se zobrazuje při kliknutí na pin dané anotace na mapě.

2.4 Návrh uživatelského rozhraní

Tato sekce podrobněji rozebírá, jaké prvky budou v aplikaci použity, jejich rozložení v rámci jednotlivých obrazovek a jejich chování. Uživatelské rozhraní (dále jen UI) musí odpovídat požadavkům na aplikaci. To ale není vše, co musí splňovat. Požadavky se musí zpracovat takovým způsobem, aby nebyly výrazně ve sporu s doporučenými pravidly pro tvorbu aplikací v iOS ([6] a [7]). Dodržení těchto pravidel zaručuje, že se uživatel bude schopný v aplikaci rychle zorientovat. Všechny aplikace musí tato pravidla v určité míře splňovat. Tím vzniklo mnoho zavedených postupů, principů a rozložení obrazovky, na která si uživatelé zvykli a se kterými v aplikacích počítají.

K návrhu UI se využívají wireframy ([22] a [23]). To jsou v podstatě skici aplikace či webu. Definují rozložení funkčních prvků na dané stránce. Nejedná se o grafický návrh, neobsahují tudíž žádné obrázky a pokud to není nutné, tak ani barvy. Wireframy je možno vytvářet více méně jakkoliv, digitálně, tužkou na papír či třeba i lepit modely. Pro účely této práce byl zvolen program Balsamiq Mockups [24]. Ten obsahuje kromě obecných prvků i několik specifických pro iOS. Nástrojů je mnoho a všechny splňují podmínky k vytvoření wireframů pro tuto práci. Balsamiq byl vybrán čistě na základě doporučení.

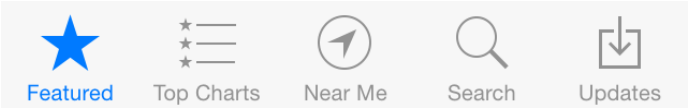
Tento návrh probíhá ještě před vývojem samotné aplikace. Návrh UI ve formě wireframů se pošle zadavateli práce, zda s daným rozložením souhlasí. Pokud návrh odpovídá požadavkům a zadavatel ho schválí, bere se tento návrh jako pevně dané zadání pro vývoj. V tento moment může začít jak implementace, tak návrh grafiky.

2.4.1 Navigace

Aplikace se skládá z několika sekcí, mezi kterými je potřeba přepínat. Jmenovitě jsou to sekce video, rádio, mapa a kvízy. Androidí verze k tomuto účelu využívala „hamburger menu“. V něm kromě těchto sekcí bylo navíc přihlášení, sdílení na sociální sítě a nastavení. Tento přístup má však několik nevýhod. Při první návštěvě aplikace je uživatel vhozen do nějaké primární sekce. V tento moment však nevidí, jak je aplikace strukturovaná, nevidí, kde se v rámci celého pavouka obrazovek aplikace nachází, a neví, jaké další možnosti má. Další nevýhoda je specifická právě pro vývoj iOS. Aplikace iOS tento navigační způsob nativně nepodporují. To je právě z důvodu, aby se zamezilo velmi jednoduchému, a právě proto lákavému vyřešení navigace v aplikaci. Nutí to

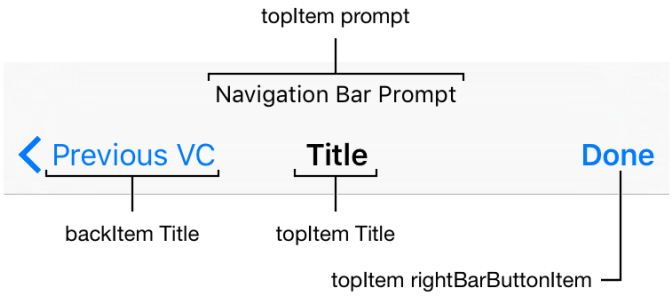
vývojáře se při tvoření UI zamyslet, jestli by se navigace po aplikaci nedala řešit nějak jinak a čistěji.

V této práci bude navigace řešena pomocí tří prvků – UITabBar, UINavigationBar a UISegmentedControl. UITabBar je spodní lišta s ikonami, mezi kterými se dá přepínat. To se hodí na přepínání mezi obrazovkami, které jsou na stejné úrovni a nesouvisí spolu. To se hodí právě pro přepínání mezi sekcemi CATVUSA aplikace. Zbylé prvky, jako jsou odkazy na webové stránky projektu, sociální sítě, přihlášení a nastavení, budou sjednoceny pod poslední sekcí, a to pod uživatelskou sekce či sekcí „další“.



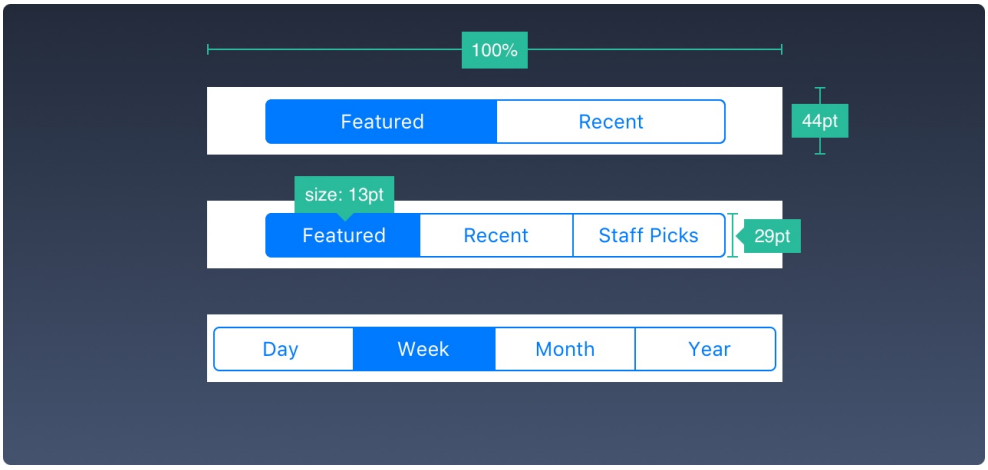
Obrázek 2.31: UITabBar [1]

V rámci jednotlivých sekcí pak bude využito UINavigationBaru, který slouží k organizaci zanořených obrazovek. Je to horní lišta, ve které často najdeme název obrazovky, na které se nacházíme, popřípadě na levé straně tlačítko zpět, pokud jsme v zanořené vrstvě navigace (třeba obrazovka nějakého detailu). Ve volném místě na liště vpravo je zvykem dávat ikonky nějakých nástrojů souvisejících s danou obrazovkou, popřípadě i nějaké popup menu, je-li jich více.



Obrázek 2.32: UINavigationBar [2]

V sekcích, kde bude potřeba přepínat mezi kategoriemi obsahu (video – broadcast, class, cooking; rádio – classic, folk) či mezi různými typy zobrazení (POI – mapa, seznam), bude využito prvku UISegmentedControl. To je několikasegmentový přepínací prvek, ve kterém je možné vybrat právě jeden prvek, nebo ve výjimečných případech žádný.



Obrázek 2.33: UISegmentedControl [3]

2.4.2 Seznam videí

Videa mají 3 kategorie – broadcast, class, cooking. Pro tyto kategorie budou existovat 3 oddělené seznamy. Mezi těmi bude možné přepínat pomocí již zmíněného UISegmentedControl. Ten se bude nacházet pod UINavigationBarem. Samotný seznam bude řešen prvkem UITableView. Ten v iOS reprezentuje list. Jeho jednotlivé buňky tvoří UITableViewCell. Ta může obsahovat malé ikony, obrázky a popisující texty. Pokud je potřeba buňka složitější, je nutné si vytvořit vlastní. Toho bude využito právě v těchto seznamech. Buňky budou tvořeny velkým náhledovým obrázkem videa, pod kterým bude krátký popis, čeho se video týká.

2.4.3 Detail videa

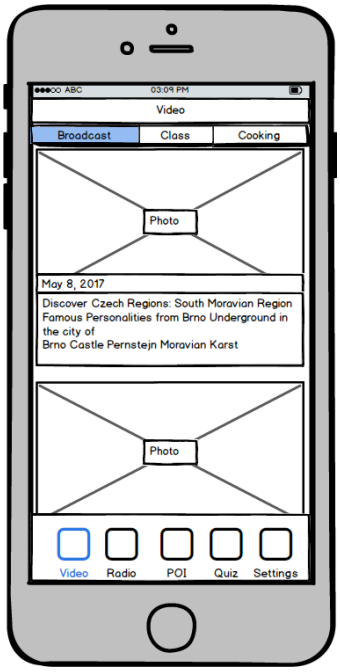
V horní polovině obrazovky bude video přehrávač. Pod ním budou doprovodné texty s případnými obrázky. V horní liště bude na rozdíl od seznamu tlačítko zpět, přes které se uživatel může vrátit na seznam.

2.4.4 Rádio

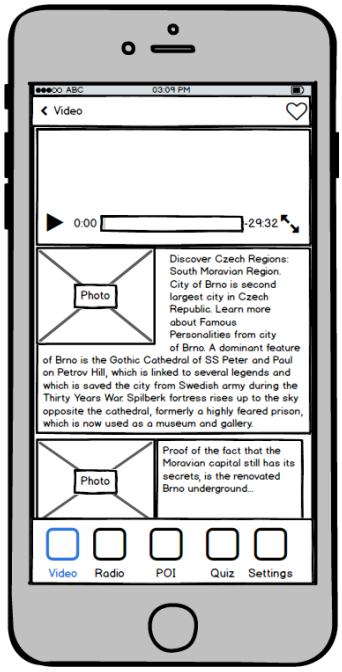
Rádio má stejně jako sekce s videi několik kategorií. Pro přepínání mezi nimi bude využit stejný princip. Ve zbytku obrazovky pak bude doprovodný obrázek písně (popřípadě logo CATVUSA, pokud žádný není), popis písně a úplně dole pak ovládání přehrávání.

2.4.5 Zajímavá místa

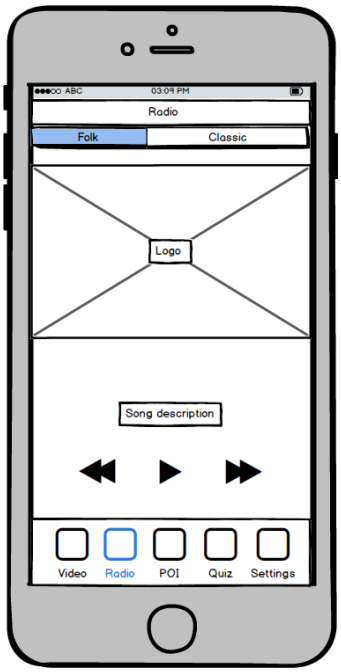
Sekce sice nemá kategorie, nicméně seznam zajímavých míst je dobré zobrazovat jak na mapě, tak ve formě seznamu. Připínání mezi zobrazeními bude



Obrázek 2.34: Video list

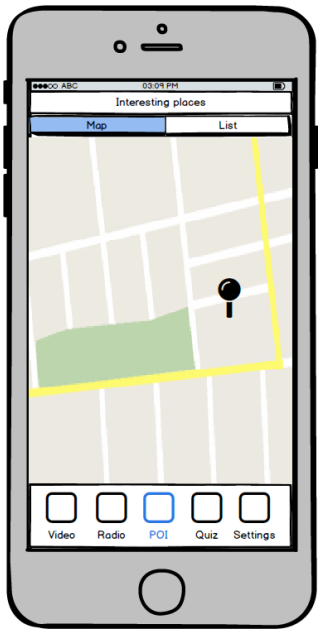


Obrázek 2.35: Video detail



Obrázek 2.36: Radio

fungovat stejně jako přepínání mezi kategoriemi u video seznamu. Mapa je vyobrazena pomocí prvku MKMapView. Seznam pak pomocí jednoduchého UITableView.



Obrázek 2.37: Map

Obrázek 2.38: Map list

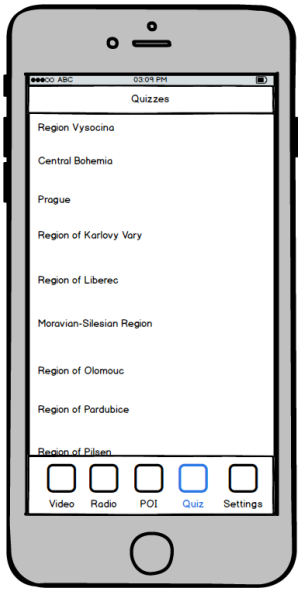
2.4.6 Kvíz

Kvízová sekce má několik obrazovek. První z nich je seznam kvízů (obrázek 2.39). Zde bude pouze seznam jednotlivých kvízů, realizovaný pomocí UITableView (seznamem). Při kliknutí na jednu z položek v seznamu se přejde k samotnému kvízu.

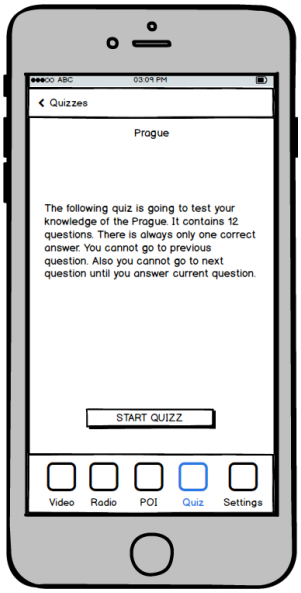
Na začátku každého kvízu je krátký popis kvízu (obrázek 2.40). Bude zde UILabel s názvem kvízu, UITextView s informacemi o délce kvízu a instrukcemi k jeho vyplnění. Pomocí tlačítka ve spodní části obrazovky bude uživatel moci kvíz spustit.

Obrazovka s otázkou (obrázek 2.41) bude obsahovat nadpis s číslem otázky, text otázky, odpovědi a doprovodný obrázek, vyjadřující stav otázky. Po zodpovězení otázky se správná odpověď vybarví zeleně a špatná červeně. Též se změní obrázek a ve spodní části obrazovky se zobrazí tlačítko, umožňující přejít k další otázce. Systém iOS nemá žádný nativní prvek vyjadřující checkbox, bude tak nutné vytvořit vlastní.

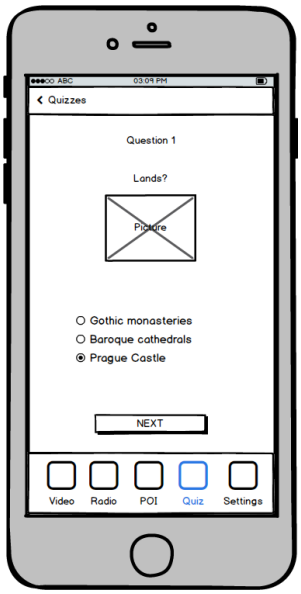
Na ukončující obrazovce (obrázek 2.42) bude pouze informace o počtu správných a špatných odpovědí.



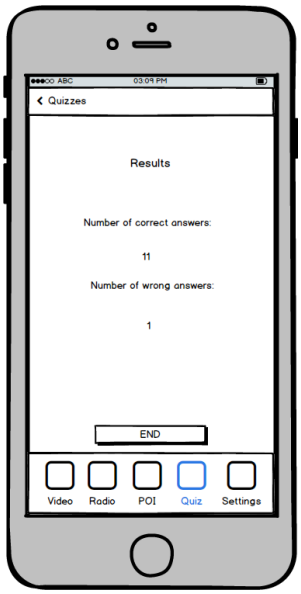
Obrázek 2.39: Quiz list



Obrázek 2.40: Quiz start



Obrázek 2.41: Quiz



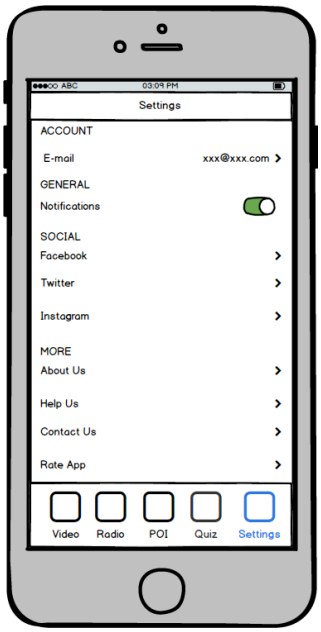
Obrázek 2.42: Quiz end

2.4.7 Uživatelská sekce

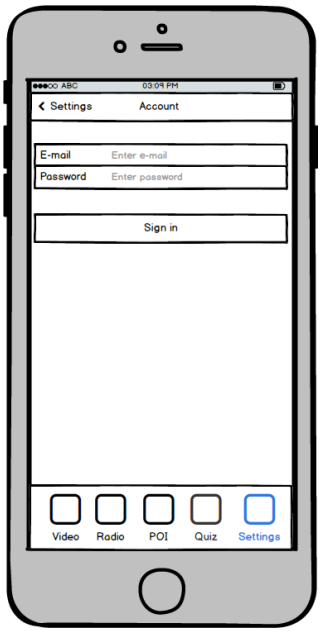
Uživatelská sekce (či „sekce další“ nebo „nastavení“) je pouze UITableView s buňkami dělenými do skupin. Tyto skupiny sdružují související buňky dohromady. Většina buněk jsou pouze jednoduché prvky UITableViewCell s odkazy.

Buňka s nastavením notifikací v sobě však obashuje UISwitch.

Další obrazovkou je přihlášení. K tomu se dostaneme při kliknutí na buňku „E-mail“ v uživatelské sekci. Zde jsou pouze dvě pole k vyplnění e-mailu a hesla a potvrzovací tlačítko. Pokud je uživatel již přihlášen, zůstane pouze pole s e-mailem a tlačítko se změní na odhlašovací.



Obrázek 2.43: Settings

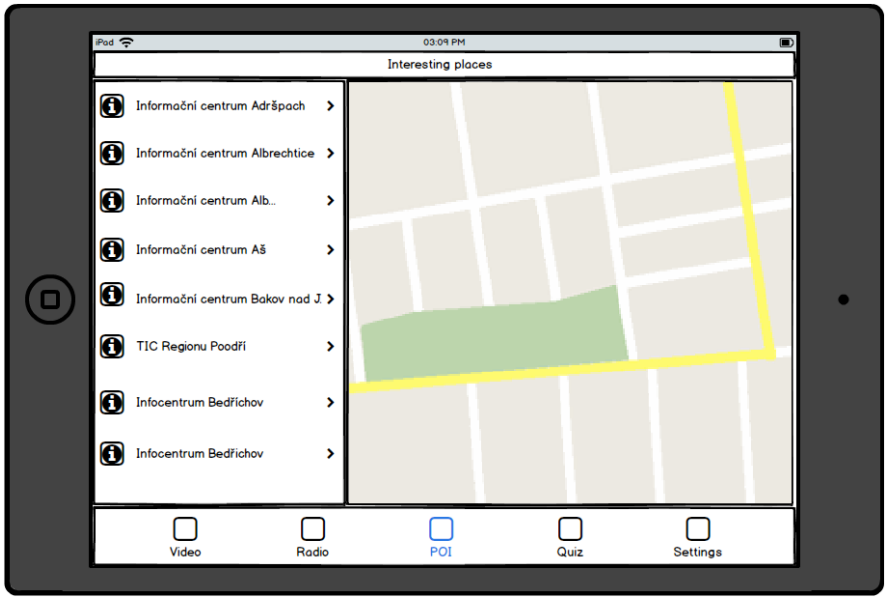


Obrázek 2.44: Sign In

2.4.8 iPad

Aplikace by měla kromě telefonů iPhone fungovat i na zařízeních iPad. Na tabletu se však ne vždy hodí stejný layout. Proto byly pro iPad provedeny následující změny. Na obrázku 2.45 můžeme vidět, že sekce „Zajímavá místa“, která původně byla rozdělena do dvou zobrazení, je nyní pouze na jedné. K prezentaci uživateli je využito UISplitView. To se skládá z levého sloupce, ve kterém je většinou UITableView, a z většího okna vpravo, ve kterém se zobrazuje detail dané položky. Levému sloupci se říká „Master view“ a pravému oknu „Detail view“.

Stejným principem jsou pak řešeny i sekce „Video“ a „Quiz“. V těchto sekcích je v „master view“ seznam (viz obrázek 2.34) a v pravém oknu je obrazovka, která by se na iPhonu zobrazila po rozkliknutí dané položky (viz obrázek 2.35).



Obrázek 2.45: iPad Map

Kapitola 3

Realizace

Tato kapitola se bude zabývat technologiemi (a frameworky) využitými při vývoji aplikace a dále vývojem samotným. Budou popsány důvody výběru použitého vývojového prostředí, jazyka, architektury a strukturování projektu.

3.1 Využité technologie a frameworky

3.1.1 Xcode

Tvorba aplikací pro iOS je velmi omezena faktem, že nástroje na vývoj fungují pouze na MacOS. Co se týče IDE, je zde Xcode [25]. Prostředí zdarma nabízené vývojářům přímo of firmy Apple. Xcode je ve většině ohledů velmi dobré IDE, nicméně má své nedostatky. Hlavním důvodem, proč vývojáři často hledají alternativu k tomuto programu, jsou zaostalé nástroje na editaci textu. Ačkoliv je Xcode udržovaný, jsou nástroje tyto z nějakého důvodu o pár let pozadu za editory jiných firem. Dalšími nedostatky jsou například nemožnost upravit vzhled a polohu nástrojových lišt editoru a pouze jednoduchá provázanost s verzovacím nástrojem Git.

Bohužel jedinou alternativou, která stojí za zmínku je AppCode [26]. Tento editor z dílny známé firmy JetBrains se snaží napravit chyby Xcode. To se JetBrains povedlo. Výše zmíněné nedostatky odstranili a v tomto ohledu AppCode vede. Proč tedy nepoužívat AppCode [27]? Důvodů je hned několik. Prvním a největším je absence Interface Builderu. To je čím dál více využívaný UI editor zabudovaný v Xcode. AppCode měl vlastní „UI designer“, nicméně bylo téměř nemožné stíhat implementovat změny, které Apple dělal v Interface Builderu, a tak byl vývoj zastaven. AppCode tak soubory „.storyboard“ otevírá přímo v Xcode. S tím je svázaný další problém většiny IDE pro iOS. Zatímco Xcode běží samostatně, ostatní editory potřebují jeho nástroje. Je tedy nutné mít nainstalované jak je, tak Xcode.

V tomto projektu bude z velké části využíván Interface Builder, a proto byl zvolen Xcode jako IDE pro vývoj. Dalšími důvody pro tuto volbu jsou lepší

zkušenosti s tímto prostředím a fakt, že vývojářská licence AppCode vyjde na necelých 200 $.

3.1.2 Swift

Pro vývoj mobilních aplikací pro iOS je možné využít mnoho různých jazyků. Vývoj v jazycích vytvořených pro tuto platformu však nabízí více možností a jednodušší řešení problémů. Proto se řešil výběr pouze ze dvou jazyků, a to Objective-C a Swift [28].

Objective-C (dále jen Obj-C) je jazyk starší, vytvořený v roce 1984, založený na jazyce C. Rozvoj tohoto jazyka tak byl omezen rychlostí rozvoje jazyka C. To byl problém, a proto začal vznikat jazyk nový – Swift. Ten byl představen v roce 2014. Swift je syntakticky mnohem více podobný například JavaScriptu. Oproti Obj-C, které využívalo závorky například i pro volání funkce, je Swift mnohem čitelnější a jednodušší.

Hlavní výhodou Swiftu je koncept takzvaných „optionals“. Jde vlastně o náhradu pointerů a zabezpečení proměnných. Přesněji se jedná o správu nil pointerů. V Obj-C, pokud voláme metodu s nil pointerem, aplikace sice nespadne, nicméně vzniká možnost nepředvídatelného chování. Swift představením „Optional types“ přesouvá zodpovědnost hlídání nil pointerů na IDE, které nenechá programátora metodu nad proměnnou zavolat, pokud ji nejdříve neošetří nebo si takové chování nevynutí.

To je též spojené s lepší podporou ARC („Automatic reference counting“). To bylo sice podporováno již Obj-C, ne však kompletně. Některé části jako například „Core graphics“ (framework zodpovědný za vykreslování vlastních tvarů a grafiky) ARC nepodporovaly a vývojář byl nucen spravovat paměť sám. Ve Swiftu je tato podpora napříč celým systémem.

Výhod Swiftu je samozřejmě mnohem více. Nemalý vliv na to má fakt, že se tento jazyk stále aktivně vyvíjí a je přihlíženo na názory vývojářů. Jediným případem, kde se stále používá Obj-C, jsou již rozjeté větší projekty, u kterých by konverze do Swiftu byla komplikovaná. Z výše uvedených důvodů byl tedy zvolen jazyk Swift, přesněji jeho aktuální verze 3.0.

3.1.3 Interface Builder

Interface Builder (dále pouze IB) je nástroj pro usnadnění tvorby UI aplikací pro platformy od Apple (MacOS, iOS a další). IB vznikal původně jako samostatná aplikace, je však součástí Xcode od verze 4.0. Ačkoliv k němu byla zprvu většina vývojářů skeptická, postupem času se uchytil. Za to může i to, že bylo v posledních letech věnováno mnoho úsilí do jeho rozvoje a optimalizace. Přibylo mnoho funkcionalit a není již potřeba takového výpočetního výkonu jako dříve.

Tvorba UI pomocí kódu má mnoho nevýhod. Je to náročné, změny nejsou vidět okamžitě, je nutné vždy projekt zkompilovat a vyzkoušet, ve

složitějších případech vznikají dlouhé metody popisující pouze layout prvků na obrazovce. Obzvláště pak v případech, kdy je potřeba rozdílný layout pro různá zařízení, to vede k chaosu zvanému „spaghetti code“. Tento výraz označuje špatně strukturovaný kód, zabalený v dlouhých metodách s mnohonásobným větvením.

Apple tento proces zkracuje tímto designovým editorem, ve kterém si uživatel může rozhraní aplikace naklikat myší a změny vidí okamžitě. To navíc šetří stovky řádků kódu. IB navíc podporuje „autolayout“ a „constraints“. Není tedy nutné UI tvořit staticky. Prvky se dynamicky přizpůsobí dle nastavení vývojáře. IB jde však ještě dále. Kromě obyčejných „view“ (oken / prvků) je možné vytvořit takzvaný „Storyboard“. Ten umožňuje vložení controllerů, vyjadřujících jednotlivé obrazovky v rámci aplikace. Tyto controllery je možné mezi sebou propojovat a vytvořit si tak jakousi mapu aplikace. Navíc lze jednotlivým prvkům obrazovky, dle jejich typu, přiřadit funkci, kterou pak stačí v kódu pouze implementovat. Celkově je provázání s vývojem v textovém editoru velmi rozvinuté.

Ačkoliv to zní všechno velmi dobře, ne každý IB však využívá. Na IB je třeba si zvyknout, je to něco velmi odlišného od tvorby UI v kódu. Proto k tomu někteří neradi přecházejí. Není to však jen o stylu tvorby aplikace. Například při využití IB mají proměnné v něm vytvořené trochu jiný „life cycle“ než při přístupu v kódu. Vývoj se tedy liší i z pohledu „workflow“ logiky.

3.1.4 Alamofire

Aplikace CATVUSA si potřebuje alespoň jednou stáhnout informace ze serveru. Ty si pak drží lokálně v zařízení a pouze si informace updatuje. Nicméně nějaká síťová komunikace zde je. K té je využit framework Alamofire.

Je to alternativa k řešení CFNetwork od Applu. To je však v mnoha ohledech složitější a zdlouhavé. Alamofire tyto problémy eliminuje zabalením tohoto kódu do logických celků. Použití se tak stává mnohem jednodušším s minimálními ztrátami na funkcionalitě. Pro jednoduchá RESTová API je zde ještě jednodušší alternativa s názvem RestKit. To je Objective-C framework založený na AFNetworking (Obj-C varianta Alamofire). Ten zjednodušuje práci se základními požadavky na REST API. Na složitější práci se síťovou komunikaci se však nehodí.

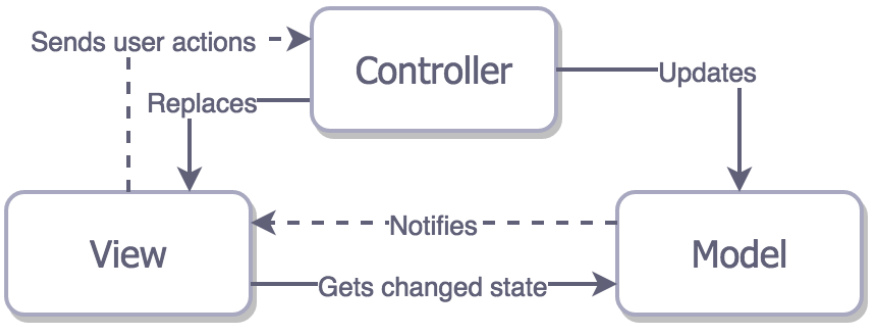
Vzhledem k předchozím zkušenostem a úrovni zanoření JSONových odpovědí serveru byl zvolen Alamofire. Mapování objektů z JSONu by se v RestKitu v tomto případě mohlo stát velmi nepříjemným zážitkem.

3.2 Architektura aplikace

Jedním z nejdůležitějších rozhodnutí před začátkem vývoje aplikace je volba architektury. Ta je závislá na rozsahu a povaze daného projektu. Například

při tvorbě jednoduché aplikace zobrazující svátky lidí v daný den je zbytečné použít složitější architekturu jako je VIPER. Kód bude zbytečně složitý, rozházený do několika souborů a hlavně to zabere mnohem více času. V následujících několika odstavcích budou rychle shrnuty nejznámnější používané typy architektur, jejich rozdíly a využití (z [4]).

První architekturou je zároveň ta nejvíce jednoduchá, a to MVC (Model-View-Controller). Z obrázku 3.1 je vidět vysoká provázanost tohoto řešení. Prezentační vrstva (View) ví o objektech reprezentujích data (Model). To je velice nepraktické, jelikož i menší změny například právě v datové vrstvě je třeba měnit napříč všemi vrstvami.

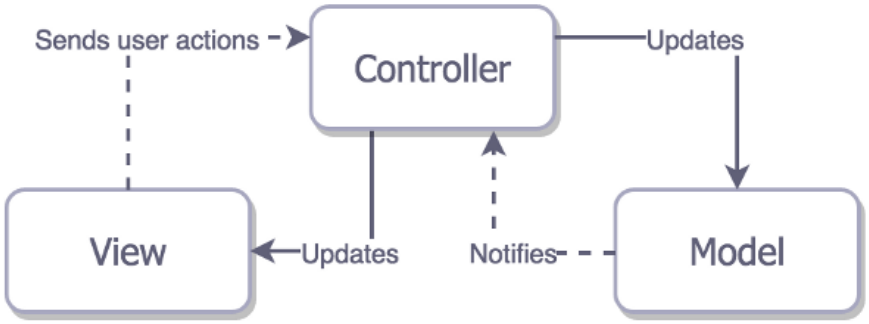


Obrázek 3.1: MVC architecture [4]

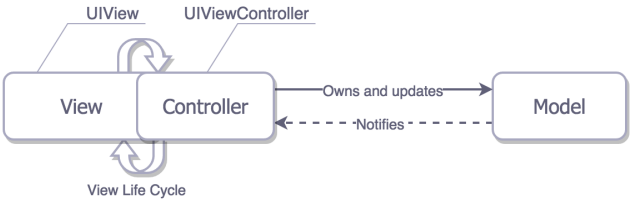
Apple ve své dokumentaci doporučuje lehce upravenou variantu MVC, a to dle obrázku 3.2. Z něj je vidět, že zmizela závislost View na Modelu. To však reálně znamená, že veškerá transformace z datových objektů na čitelná data byla přesunuta do Controlleru, který má přístup k oběma vrstvám. Tím se sice snížila provázanost a projekt získal na modularitě, jelikož View a Modely jsou nyní v podstatě samostatné. Nicméně utrpěla vrstva Controllerů, které jsou nyní zodpovědné za všechno. Takovým controllerům se říká „Massive View Controllers“. Tento problém je velmi diskutovaný ve světě mobilního vývoje a častým tématem je, jak tuto zodpovědnost přesunout z controlleru jinam.

Dalším problémem MVC od Applu je tendence propojit View a Controller v nerozlučnou vrstvu jako na obrázku 3.3. To má za následek ještě chaotičtější controller a časté porušení MVC, kdy je datový objekt controllerem předán přímo do View, které si ho zpracuje.

První z architektur, které se tyto problémy snaží řešit je MVP (Model-View-Presenter). Na první pohled je tato architektura velmi podobná upravené variantě MVC od Apple (jak je vidět na obrázku 3.4). To však není tak úplně pravda. Zatímco v MVC je View velmi úzce svázané s Controllerem, v MVP je Presenter zodpovědný pouze za předávání dat a správu stavu View. View vrstvu zde tvoří View a ViewController dohromady. Presenter se k nim chová

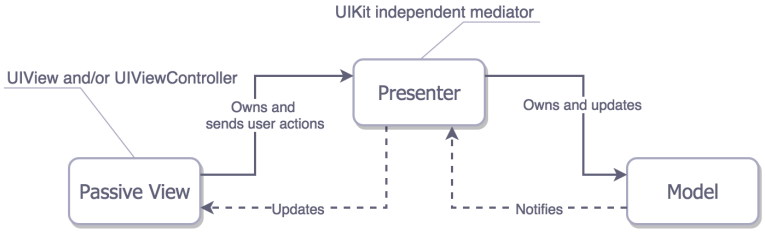


Obrázek 3.2: Apple's MVC [4]



Obrázek 3.3: Apple's MVC in practice [4]

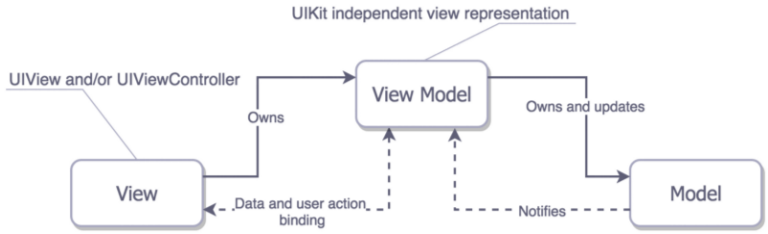
jako k View. Tohle View si však samo spravuje například layout.



Obrázek 3.4: MVP architecture [4]

Další architekturou je MVVM (Model-View-ViewModel). Ta ještě trochu upravuje MVP. Na obrázku (3.5) však vypadá skoro stejně. Rozdíl je v provázanosti View a ViewModelu. Zatímco Presenter pouze data dodává, ViewModel je s View provázán a navzájem tudíž ví o jakékoliv změně, která nastane. MVVM je vhodné už i pro větší projekty, nicméně je častou praxí, že si vývojáři tuto architekturu přizpůsobují k potřebám projektu. Výhodou MVVM je to, že se ze správně implementovaného MVC na tuto architekturu dá plynule přejít.

VIPER [29] je architektura založená na principu jedné odpovědnosti, a tím se od ostatních výrazně liší. Navíc se zodpovědnosti rozdělily ze 3 vrstev do 5. Těmi jsou View-Interactor-Presenter-Entity-Router (obrázek 3.6).



Obrázek 3.5: MVVM architecture [4]

- **View** – Vrstva se skádá jak z View, tak z ViewControlleru. Jejím úkolem je prezentovat data a předávat uživatelské akce Presenteru.
- **Interactor** – Je naprosto nezávislý na jakémkoliv UI. Spravuje entity a vykonává specifický use case. Entity nikdy nepředává do Presenteru, ten dostane už jen jednoduché datové struktury.
- **Presenter** – Má na starosti 3 úkoly. Reagovat na uživatelskou interakci z View, říkat Routeru, kam navigovat, a požadovat data od Interactoru.
- **Entity** – Objektové reprezentace dat, získané z nějakého zdroje. S entitami pracuje pouze Interactor.
- **Router** – Zatímco Presenter má informace o tom, kdy a co zobrazovat za obrazovku, router ví, jak se tam dostat. Spravuje tak navigaci v rámci aplikace.

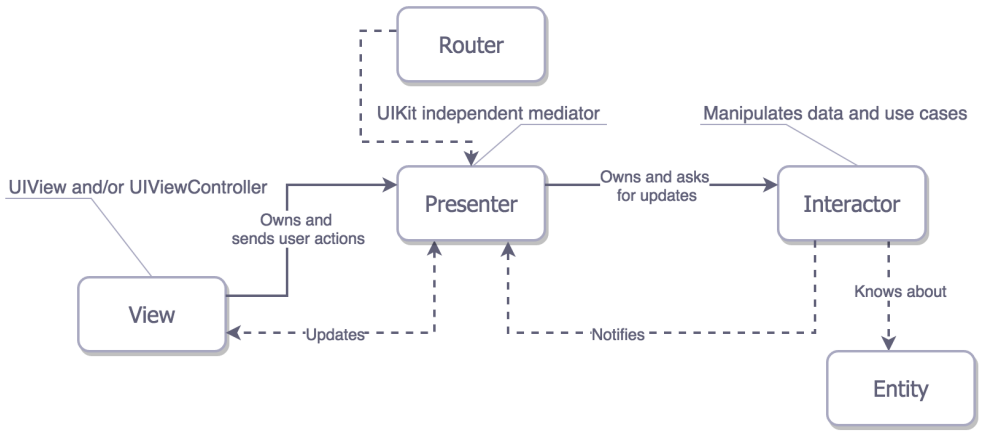
Takováto pětice vrstev dohromady tvoří jeden modul. Modul reprezentuje jednu nebo několik souvisejících obrazovek v rámci aplikace. Záleží tak pouze na vývojáři, jak velké moduly budou. Tím, že jsou úlohy takto podrobně rozdělné, je jednoduché jednotlivé části nahrazovat či testovat.

VIPER se hodí spíše na větší projekty či projekty, které počítají s budoucím rozvojem. O využití této architektury je však třeba rozhodnout předem, jelikož přechod z libovolného MV(X) na VIPER by byl velmi složitý. Implementace této architektury zabere mnoho času ze začátku vývoje, proto se nehodí pro menší projekty, které spíše zdržuje.

V této práci bude využita architektura MVC s úpravami doporučenými Applem. Projekt není v aktuálním stavu tak rozsáhlý, aby bylo zapotřebí složitějších architektur. Bude však přihlíženo na možnost rozvoje aplikace a kód bude strukturován tak, aby bylo možné přejít na architekturu MVVM.

3.3 Struktura aplikace

Tato sekce popisuje, jak se v rámci implementace strukturovaly soubory a kód. Obrázek 3.7 je screenshot této struktury přímo z Xcode.



Obrázek 3.6: VIPER architecture [4]

3.3.1 Core

Základ celé aplikace. Obsahuje zdrojové kódy všech obrazovek v aplikaci. Ty jsou rozděleny do složek podle již známých sekcí aplikace. Každá z těchto složkek odpovídá architektuře MVC. To je vidět na sekci „Quiz“ ve screenshotu 3.7.

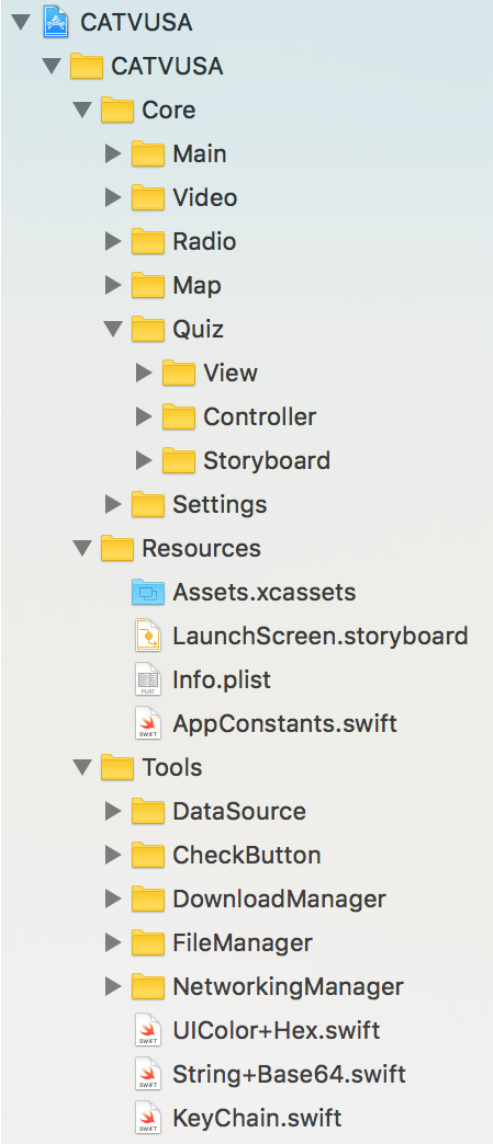
V podsložce „Storyboard“ jsou pouze soubory .storyboard, které obsahují všechny obrazovky dané sekce.

Podsložka „Controller“ obsahuje implementace navržených controllerů využitých ve storyboardech. Tato složka též obsahuje třídy (struktury) modelové vrstvy. Není jich mnoho, a tak jsou vždy ve stejném souboru jako controller, se kterým jsou svázány.

V poslední podsložce nazvané „View“ jsou soubory .xib, obsahující dodatečné prvky vrstvy View. Ty jsou navrženy v Interface Builderu a následně doimplementovány v kódu v souborech s identickým jménem (avšak koncovkou .swift).

3.3.1.1 Main

Tato sekce se ostatním na této úrovni vymyká. Obsahuje totiž soubory využité při spuštění aplikace. Přesněji AppDelegate a storyboardy rozdělující zobrazení na iPhone a iPad. AppDelegate je třída implementující metody, které jsou volané při různých interakcích aplikace a systému. Například při spuštění aplikace, přesunu aplikace do pozadí či jejím ukončení. Storyboardy v této sekci implementují pouze hlavní navigační prvek, a tím je UITabBar.



Obrázek 3.7: Project structure

3.3.1.2 Video

Obsahuje implementaci první sekce – video. Tu tvoří 4 controllery, a to VideoMasterViewController, VideoMenuViewController, VideoTableViewController a VideoDetailViewController. Kromě těch je zde ještě View vrstva, obsahující 2 třídy dědící od UITableViewCell. Jsou to upravené buňky pro tabulky využité v této sekci.

První jmenovaný controller je podtřídou UISplitViewController. Je to pouze navigační controller, který se stará o zobrazení správných detailů vi-

dea, dle uživatelské interakce. Obsahuje též UINavigationController, který je využitý k orientaci mezi jednotlivými obrazovkami.

VideoMenuViewController je též pouze navigační controller, zajišťující přepínání mezi jednotlivými kategoriemi video seznamů. K tomu využívá prvek UISegmentedController.

Dalším controllerem je VideoTableViewController. Ten reprezentuje seznam videí v rámci jedné kategorie. Implementuje veškerou logiku třídy UITableView. Též má na starosti načtení dat skrze třídu DataSource (více v sekci 3.3.3.1).

Posledním controllerem v této sekci je VideoDetailViewController. Ten je zodpovědný za přehrávání videa a zobrazení jeho kapitol pod přehrávačem v přehledném seznamu. K tomu využívá AVPlayerViewController a UITableView.

3.3.1.3 Radio

Implementace této sekce obsahuje oproti videu pouze dva controllery. Těmi jsou RadioMenuViewController a RadioDetailViewController. Zvlášť je zde oddělena modelová vrstva (soubor RadioList), která zde není pouhou čistě datovou strukturou, nýbrž má v sobě i vlastní logiku (sekce 2.3.2 – červená).

RadioMenuViewController má, stejně jako VideoMenuViewController, na starost navigaci mezi dvěma RadioDetailViewControllery. K jejich přepínání též využívá UISegmentedControlller. Další jeho zodpovědností je načtení dat z DataSource a následné předávání specifických údajů controllerům s detailem.

Zbývající třída RadioDetailViewController pak zodpovídá za přehrávání písní pomocí třídy AVPlayer a zpracování uživatelské interakce s tlačítky přehrávače.

3.3.1.4 Map

Tato sekce je strukturovaná podobně jako předchozí. Jsou zde však 3 rozdílné controllery. MapMenuViewController, MapViewController a MapTableViewController. Mimo těch je tu ještě třída MapPoiTableViewCell, která dědí od UITableViewCell. Je to vzhledově upravená buňka pro UITableView využité v této sekci.

MapMenuViewController má za úkol přepínat mezi instancemi controllerů MapViewController a MapTableViewControlleru. Dále pak ještě zpracovává jakékoliv transakce mezi těmito instancemi. Jelikož tyto controllery nezobrazují rozdílný obsah, nýbrž ten stejný v různých podobách, existuje mezi nimi komunikace alespoň na základní úrovni (například zobrazit záznam ze seznamu na mapě). Controllery však o sobě navzájem neví, komunikace je proto zpracovávána v tomto jejich rodičovském controlleru. Třída má také

zodpovědnost za načtení dat pomocí DataSource a předání do svých potomků (zmíněné zobrazující controllery).

První z controllerů znázorňujících data je MapViewController. Ten zobrazuje jednotlivé záznamy pomocí pinů na mapě. To je jediná zodpovědnost tohoto controlleru. Jsou zde implementovány metody určující, které záznamy se mají zobrazit, kde a v jaké formě. To zahrnuje i algoritmus na slučování pinů z určité oblasti, aby mapa nebyla přehlcena záznamy.

Posledním controllerem je MapTableViewController, který stejná data zobrazuje ve formě seznamu. K tomu využívá již zmíněné buňky MapPoiTableViewCell.

3.3.1.5 Quiz

Quiz je rozdělen do tří částí – Navigace (QuizMasterViewController), seznam kvízů (QuizMenuViewController) a kvíz samotný (QuizDetailViewController). Mimo controllery jsou zde ještě 3 .xib soubory. Tato View znázorňují stavy kvízu – úvodní obrazovka kvízu (QuizStartView), obrazovka s otázkou (QuizQuestionView) a finální obrazovka s vyhodnocením (QuizFinishView).

QuizMasterViewController dědí od UISplitViewControlleru. Slouží pouze jako zapouzdření pro rozdílné layouty na iPhonu a iPadu.

QuizMenuViewController využívá jednoduché UITableView. Je zodpovědný pouze za načtení dat z DataSource a jejich seznamové zobrazení.

QuizDetailViewController je již zajímavější než předcházející controllery. Jeho úkolem je zobrazit kvíz s daty dodanými MenuControllerem. K tomu využívá výše zmíněné View. Dle toho, kde se v kvízu uživatel nachází, zobrazuje rozdílné View.

3.3.1.6 Setting

Implementace uživatelské sekce obsahuje 3 controllery, a to SettingViewController, SettingAccountViewController a SettingWebViewController. Dále se zde nachází LoggingView, prezentované při procesu přihlašování uživatele, a dvě upravené buňky pro UITableView (například buňka s UITextField – s textovým polem pro e-mail a heslo při přihlášení).

SettingViewController je jednoduchý controller, který využívá statické UITableView. Počet buněk seznamu ani jejich vzhled se nemění, lze jej proto vytvořit v Interface Builderu. Kódová implementace controlleru pak spravuje uživatelskou interakci s tímto seznamem. Ten tvoří odkazy na webové stránky, do AppStore, nastavení notifikací a přesměrování na obrazovku s přihlášením.

SettingAccountViewController znázorňuje stránku s přihlášením. Ta je ve formě upraveného dynamického UITableView. Zobrazuje specifické buňky podle toho, zda je uživatel přihlášen či odhlášen. Též zodpovídá za komunikaci se serverem (ohledně přihlášení) pomocí třídy NetworkingManager (více v sekci 3.3.3.2).

SettingWebViewController znázorňuje obrazovku, která se dostane na popředí, když uživatel rozklikne nějaký odkaz v seznamu v SettingViewControlleru. Pro znázornění této webové stránky se využívá UIWebView.

3.3.2 Resources

Složka pro ukládání statického obsahu aplikace. Aktuálně obsahuje 4 soubory, jak je vidět na obrázku 3.7. Soubor Assets.xcassets obsahuje obrázky k projektu (logo a ikony). LaunchScreen.storyboard je speciální storyboard, jehož obrazovka se zobrazí při spuštění aplikace, přesněji mezi kliknutím na ikonu aplikace a načtením první obrazovky. Info.plist je XML soubor obsahující seznam hodnot různých typů uložených pod danými klíči. Obsahuje globální nastavení aplikace, jako jsou například verze buildu, povolená natočení obrazovky, minimální verze systému a další. Posledním souborem je AppConstants. Ten jako jediný z nich byl vytvořen manuálně. Obsahuje několik tříd se statickými proměnnými. Jsou to konstanty vztahující se ke specifickým funkcím aplikace. Například různé identifikátory, konstantní texty a globální barvy projektu.

3.3.3 Tools

Aplikace obsahuje mnoho kódu, který nepatří do žádné specifické sekce. Tyto třídy jsou dostupné globálně ať už ve formě singletonu (jediná instance třídy pro celou aplikaci), nebo jako třída s pouze statickými metodami. Přesněji třídními metodami. Ve swiftu je rozdíl mezi klíčovými slovy „class“ a „static“, a to v tom, že třídním metodám mohou podtřídy přepsat implementaci. Tyto nástroje jsou nezbytné pro běh aplikace.

Existují zde ještě další dva typy „pomůcek“, kterým se úplně nedá říkat nástroje. Prvním je například CheckButton, což je v podstatě UIView (více v sekci 3.3.3.6). Druhým jsou pak rozšíření tříd takzvané „extensions“. Ty mohou rozšířit jakoukoliv třídu o novou metodu či funkcionality. Příkladem je například rozšíření třídy UIColor (třída systémového frameworku UIKit) o třídní metodu, která vrátí instanci vytvořenou z hexadecimální reprezentace barvy (ukázka 3.1).

Listing 3.1: UIColor extension

```
import UIKit

extension UIColor {

    class func from(hexNumber: Int) -> UIColor {
        ...
    }
```

```
}
```

3.3.3.1 DataSource

Nástroj DataSource slouží jako zdroj dat pro celou aplikaci. Je to jakýsi zprostředkovatel mezi controllery a síťovým managerem (NetworkingManager – viz 3.3.3.2). DataSource má narozdíl od NetworkingManageru přístup k modelové vrstvě aplikace. Controllery ji tak žádají již o přesné datové typy, které pak využívají. Výhodou tohoto nástroje je možnost odtrhnout NetworkingManager a připojit jiný zdroj dat. Což je zde zároveň využito i pro cachování. DataSource si odpovědi od serveru ukládá, a pokud je aplikace offline, využije poslední uloženou odpověď. Controllery, které většinou složitě řeší případy, co se děje v různých stavech aplikace, o tomto problému vůbec netuší a pouze zobrazují dodaná data.

Součástí nástroje DataSource je také třída JSONParser, která se stará o transformaci JSONové odpovědi serveru na aplikaci známé datové typy. Tyto transformace jsou z hlediska délky kódu velmi objemné a často bývají součástí controllerů, což vede k takzvaným „Massive view controllers“. Tato třída přesouvá tento problém jinam.

3.3.3.2 NetworkingManager

Síťový manager, singleton, implementující asynchronní metody komunikující se serverem CATVUSA. Slouží v podstatě jako modul spojující framework Alamofire a jádro aplikace. Pokud by v budoucnu vznikla potřeba Alamofire nahradit, bude stačit přepsat implementaci těchto metod. Framework nebude propletený celou aplikací, bude využíván pouze tímto nástrojem.

3.3.3.3 DownloadManager

Pomocný nástroj k třídě NetworkingManager (dále jen NM). Ten obsahuje právě jednu instanci této třídy. Využívá ji ke stahování souborů (v tomto případě obrázků) a správě těchto transakcí. Metody třídy DownloadManager jsou od NM odděleny, jelikož se v mnoha věcech liší. Tato spojení trvají delší dobu, je tedy třeba si držet reference na ně v případě, že je potřeba s nimi dále pracovat. K tomu je samozřejmě potřeba několik dalších metod na manipulaci těchto spojení. Kromě klasické odpovědi ze serveru je výstupem těchto metod také soubor. Ten je potřeba někam uložit, liší se tedy i logika zpracování odpovědi. Proto byla tato ucelená sada metod od NM odtržena. Je však stále od aplikace oddělena vrstouv NM, která slouží jako prostředník.

3.3.3.4 FileManager

Sada třídních metod určených pro práci se soubory. Metody jsou rozdělené do dvou částí. První jsou metody pro práci s položkami v souborovém systému aplikace. Přesněji jsou to metody, které vrací cesty k určitým složkám v aplikaci. Například složka s uloženými obrázky či složka s cachí. Dále pak metody na mazání či přístupu k těmto souborům.

Druhou částí FileManageru jsou metody pro práci s NSUserDefaults. To je systém ukládající základní nastavení aplikace. Funguje na principu Key-Value. Používá se převážně k ukládání necitlivých informací jako je uživatelské nastavení aplikace. K ukládání citlivějších informací slouží například KeyChain popsaný v 3.3.3.5. Metody v této části jsou buď typu „save“ k ukládání různých datových typů, nebo „load“ k jejich načítání.

3.3.3.5 KeyChain

KeyChain je sada třídních metod umožňujících bezpečné ukládání dat na disk. Uložit je možné jakkoliv složitou strukturu či třídu. Jedinou podmínkou je, že musí odpovídat protokolu NSCoding. Data objektu, který je potřeba uložit, jsou nejprve zakódována a následně uložena na disk pod určitým klíčem. Data uložená tímto způsobem však přetrvávají v zařízení i po odstranění aplikace (zmizí pouze s resetováním zařízení). Je proto vhodné ukládat pouze nejnutnější informace tímto způsobem. To jsou například hesla, tokeny a podobné.

3.3.3.6 CheckButton

Tento „nástroj“ se od ostatních liší tím, že je pasivní. Jsou to vlastně pouze podtřídy UIView, které se vyznačují tím, že doplňují v iOS chybějící „Checkbox“ a „Radio button“ do projektu. Třídy jsou napsané univerzálně a nezávisle na zbytku aplikace. Jsou tedy přenositelné z projektu do projektu.

Kapitola 4

Testování

Důležitou součástí vývoje aplikace je testování. Jeho účelem je objevit problémy a chyby vzniklé během implementace. Dalším přínosem je uživatelský feedback k uživatelskému rozhraní, a vůbec celkové použitenosti aplikace. Testování probíhá ještě před uvolněním aplikace pro veřejnost.

Při testování této aplikace byli využiti tři lidé z různých oborů. Před začátkem testování byli seznámeni s projektem a byla jim představena webová stránka projektu. Stejně tak, jak by k tomu přišel jakýkoliv jiný uživatel – cílovou skupinou aktuální verze aplikace jsou lidé s povědomím o projektu. Testující k aplikaci více instrukcí nedostali a měli volnou ruku v prozkoumávání rozhraní a funkcí. Průběžně aplikaci komentovali a jejich poznatky a připomínky byly zaznamenány. Tyto problémy a jejich řešení jsou popsány v následující sekci.

4.0.1 Problémy

Hlavním problémem bylo padání aplikace při rotaci mapy. To bylo způsobeno změnami v souřadnicovém systému při otáčení. Při původní implementaci algoritmu na sjednocení pinů s tím nebylo počítáno a některé proměnné tak dosahovaly záporných hodnot. Díky tomu zde vznikla nekonečná smyčka, která nakonec zahltila paměť telefonu, a systém násilně aplikaci ukončil. Naštěstí byl tento problém jediným své povahy a v jiných situacích aplikace nepadala.

Dalším problémem bylo přehrávání videa, které neskončilo, když člověk z dané obrazovky odešel. Video se dále přehrávalo kdesi na pozadí, uživatel se k němu však nemohl ani vrátit. Problém byl způsoben tím, že UISplitViewController, který menu a detail spravuje, si i po návratu z detailu tento controller držel v paměti. Podobný problém se vyskytl i u rádia, které nepřestalo přehrávat při přechodu do jiné sekce. To není nutně problémem, nicméně rádio hrálo dále i při spuštění videa, kdy pak aplikace měla dvojitý audio výstup. Oba problémy byly vyřešeny.

Posledním závažnějším problémem byly chybějící popisy u některých

videí. Byla to videa z kategorie „class“ a „cooking“, kde tyto popisy byly z API předávány pod jiným klíčem. Bylo tak nutné upravit parsovací algoritmus zpracovávající tuto odpověď.

V rámci testování vzniklo ještě okolo 10 dalších připomínek. Byly to však menší chyby jako například chybějící nadpis nebo grafické úpravy. Chyby byly opraveny a návrhy na grafické změny konzultovány se zadavatelem a případně zaimplementovány. Mezi ty patří například odstranění popisků ikon z TabBaru.

4.0.2 Závěr

Testování bylo úspěšné, objeveno bylo mnoho problémů a nedostatků, a to i přesto, že testovacích subjektů nebylo mnoho. Vzhledem k rozsahu aplikace byl však počet dostačující. Vzniklo i několik návrhů na změnu UI, které však byly po konzultaci zavrženy. Jedním z nich bylo například „Hamburger menu“ místo TabBaru. Zrovna tento návrh bude v budoucnu určitě znovu zvážen z důvodu přibývajících funkcí a sekcí.

Závěr

Jako první proběhla analýza „konkurenčních“ aplikací. Tedy aplikací s podobnou tématikou – z větší části aplikace sloužící jako průvodce. Ty byly jakousi předlohou pro to, co použít a čemu se vyvarovat. A to jak z pohledu funkcí, ale i vzhledu. Dále pak proběhla analýza požadavků, kde bylo zjištěno, jaké jsou možnosti pro řešení těchto problémů, jejich výhody a nevýhody. Z těch pak bylo vybráno to nejvhodnější pro tento typ aplikace. Následovala analýza již hotového webového API, které se ukázalo jako velmi slabý článek projektu. Byly navrženy změny, kterými by se nalezené problémy vyřešily. Dále byl vytvořen návrh datové reprezentace dat dodaných z API a uživatelské rozhraní aplikace. Zde bylo nejvíce využito informací z analýzy konkurence. Následoval výběr frameworků pro vývoj, výběr architektury a samotná implementace. Na závěr proběhlo testování aplikace a závažné chyby byly opraveny. Další připomínky byly zváženy, probrány se zadavatelem a případně implementovány.

Zadání práce bylo splněno. Aplikace slouží jako alternativa k webové části projektu v rozsahu specifikovaném zadavatelem. Aplikace je funkční a dostupná na AppStoru. Jediným nesplněným požadavkem jsou notifikace, kde je nutné počkat na implmentaci na straně serveru.

V rámci projektu probíhala častá komunikace se zadavatelem projektu a i přesto, že požadavky již byly pevně stanovené z vývoje verze pro Android, přicházely stále nové nápady, jak projekt rozšířit. Již během vývoje této verze aplikace přibyly na webu nové sekce a další budou přibývat. Vývoj iOS aplikace zde tedy rozhodně nekončí.

Aplikace má rozhodně mnoho prostoru pro budoucí rozvoj. V první řadě bude kladen důraz na zlepšení aktuálního webového API, na kterém je aplikace závislá. Dále pak na zlepšení sekce mapy, kde je potřeba sehnat nové zdroje informací a následně bude možné tuto sekci rozšířit více ve stylu ostatních průvodcovských aplikací. Tím může být i propojení s hodinkami Apple Watch.

Literatura

[1] APPLE INC.: *Tab Bars [online].* [2017-06-18]. Dostupné z: https://developer.apple.com/library/content/documentation/UserExperience/Conceptual/UIKitUICatalog/UITabBar.html

[2] APPLE INC.: *UINavigationBar [online].* [2017-06-18]. Dostupné z: https://developer.apple.com/documentation/uikit/uinavigationbar

[3] *Designing for iOS 10 [online].* [2017-06-18]. Dostupné z: https://designcode.io/iosdesign-guidelines

[4] *iOS Architecture Patterns [online].* [2017-06-23]. Dostupné z: https://medium.com/ios-os-x-development/ios-architecture-patterns-ecba4c38de52

[5] APPLE INC.: *Media Layer [online].* [2017-06-11]. Dostupné z: https://developer.apple.com/library/content/documentation/Miscellaneous/Conceptual/iPhoneOSTechOverview/MediaLayer/MediaLayer.html

[6] APPLE INC.: *iOS Human Interface Guidelines [online].* c2016, [2017-06-06]. Dostupné z: https://developer.apple.com/ios/human-interface-guidelines/

[7] APPLE INC.: *App Store Review Guidelines [online].* c2016, [2017-06-06]. Dostupné z: https://developer.apple.com/app-store/review/guidelines/

[8] *Google Trips [online].* [2017-06-09]. Dostupné z: https://itunes.apple.com/us/app/google-trips-travel-planner/id1081561570

[9] *Material Design for Android [online]*. [2017-06-09]. Dostupné z: https://developer.android.com/design/material/index.html

[10] *LiveTrekker [online]*. [2017-06-09]. Dostupné z: https://itunes.apple.com/us/app/livetrekker/id550129756

[11] *Ascape VR: Travel App - 360° World Traveler [online]*. [2017-06-09]. Dostupné z: https://itunes.apple.com/us/app/ascape-vr-travel-app-360-world-traveler/id1062915822

[12] *Czech Republic Travel Guide by Triposo [online]*. [2017-06-09]. Dostupné z: https://itunes.apple.com/us/app/czech-republic-travel-guide-by-triposo-featuring-prague/id469358123

[13] *Guides by Lonely Planet [online]*. [2017-06-09]. Dostupné z: https://itunes.apple.com/US/app/id1045791869

[14] *CzechTourism Apps [online]*. [2017-06-09]. Dostupné z: https://itunes.apple.com/cz/developer/czechtourism/id722179625

[15] Drdlíček, M.: *Aplikace pro mobilní telefony a chytré televizory pro neziskovou TV*. Diplomová práce, České vysoké učení technické v Praze, Fakulta informačních technologií, 2017.

[16] APPLE INC.: *MPMoviePlayer [online]*. [2017-06-11]. Dostupné z: https://developer.apple.com/documentation/mediaplayer/mpmovieplayercontroller

[17] APPLE INC.: *AVPlayer [online]*. [2017-06-11]. Dostupné z: https://developer.apple.com/documentation/avfoundation/avplayer

[18] *ffmpeg [online]*. [2017-06-11]. Dostupné z: https://ffmpeg.org

[19] APPLE INC.: *VideoToolbox [online]*. [2017-06-11]. Dostupné z: VideoToolboxhttps://developer.apple.com/documentation/videotoolbox

[20] *Video Toolbox and Hardware Acceleration [online]*. [2017-06-11]. Dostupné z: https://www.objc.io/issues/23-video/videotoolbox/

[21] *WordPress REST API (Version 2) [online]*. [2017-06-15]. Dostupné z: http://v2.wp-api.org

[22] *What is wireframing? [online]*. [2017-06-06]. Dostupné z: http://www.experienceux.co.uk/faqs/what-is-wireframing/

[23] *Getting Started with Wireframes [online]*. [2017-06-15]. Dostupné z: https://www.codementor.io/nicolesaidy/getting-started-with-wireframes-du107vuh7

[24] *Balsamiq Mockups 3 [online]*. [2017-06-15]. Dostupné z: https://balsamiq.com

[25] *Xcode [online]*. [2017-06-19]. Dostupné z: https://developer.apple.com/xcode/ide/

[26] *AppCode [online]*. [2017-06-19]. Dostupné z: https://www.jetbrains.com/objc/

[27] *I have tried switching from Xcode to AppCode for iOS programming [online]*. [2017-06-19]. Dostupné z: https://optionalbits.com/i-have-tried-switching-from-xcode-to-appcode-for-ios-programming-a2993d076361

[28] *Swift vs. Objective-C [online]*. [2017-06-19]. Dostupné z: http://www.infoworld.com/article/2920333/mobile-development/swift-vs-objective-c-10-reasons-the-future-favors-swift.html

[29] *Architecting iOS Apps with VIPER [online]*. [2017-06-19]. Dostupné z: https://www.objc.io/issues/13-architecture/viper/

Příloha A

Seznam použitých zkratek

CATVUSA Czech-American TV

USA United States of America

ČR Česká republika

UX User Experience

API Application Programming Interface

UI User Interface

DRM Digital Rights Management

REST Representational State Transfer

JSON JavaScript Object Notation

XML Extensible Markup Language

URL Uniform Resource Locator

POI Point of interest

HTML Hypertext Markup Language

ID Identifier

IDE Integrated Development Environment

IB Interface Builder

ARC Automatic Reference Counting

OS Operating System

MVC Model-View-Controller

MVP Model-View-Presenter

MVVM Model-View-ViewModel

VIPER View-Interactor-Presenter-Entity-Router

Příloha **B**

Obsah přiloženého CD

- `readme.txt` ..... stručný popis obsahu CD
- `src`
  - `impl` ..... zdrojové kódy implementace
  - `thesis` ..... zdrojová forma práce ve formátu LaTeX
- `text` ..... text práce
  - `thesis.pdf` ..... text práce ve formátu PDF
  - `guide.pdf` ..... instalační příručka