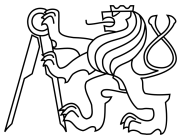
ČESKÉ VYSOKÉ UČENÍ TECHNICKÉ V PRAZE
FAKULTA INFORMAČNÍCH TECHNOLOGIÍ



ZADÁNÍ BAKALÁŘSKÉ PRÁCE

Název:	Aplikace KOSeek pro iOS
Student:	Alzhan Turlybekov
Vedoucí:	Ing. Josef Gattermayer
Studijní program:	Informatika
Studijní obor:	Softwarové inženýrství
Katedra:	Katedra softwarového inženýrství
Platnost zadání:	Do konce letního semestru 2016/17

Pokyny pro vypracování

Cílem práce je vytvoření mobilní aplikace "KOSeek" pro iOS. Aplikace bude sloužit studentům ČVUT FIT a FEL k vyhledávání kontaktů ve škole, informacím o studijním oddělení, o předmětech a harmonogramu školního roku.
Pokyny:
Proveďte analýzu původní aplikace pro Android.
Proveďte analýzu požadavků studentů ČVUT FIT a FEL.
Proveďte analýzu alespoň jedné aplikace pro jiný školní informační systém.
Prozkoumejte možnosti KOSapi.
Navrhněte wireframy aplikace a po dohodě s vedoucím zajistěte grafický návrh (grafický návrh není součástí práce).
Implementujte aplikaci pro iOS.
Proveďte testování na vybrané skupině uživatelů.

Seznam odborné literatury

Dodá vedoucí práce.

L.S.

Ing. Michal Valenta, Ph.D.
vedoucí katedry

prof. Ing. Pavel Tvrdík, CSc.
děkan

V Praze dne 17. listopadu 2015

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

Bakalářská práce

Aplikace KOSeek pro iOS

Alzhan Turlybekov

Vedoucí práce: Ing. Josef Gattermayer

8. května 2016

Poděkování

Rád bych poděkoval svému vedoucímu Ing. Josefu Gaterrmayerovi za cenné rady a připomínky při vedení mé bakalářské práce. V neposlední řadě bych chtěl poděkovat svému učiteli Swiftu Ing. Dominiku Veselýmu.

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 8. května 2016 .....

Odkaz na tuto práci

Turlybekov, Alzhan. *Aplikace KOSeek pro iOS*. Bakalářská práce. Praha: České vysoké učení technické v Praze, Fakulta informačních technologií, 2016.

Abstrakt

Tato bakalářská práce popisuje vytvoření mobilní aplikace KOSeek pro iOS. Aplikace slouží studentům k vyhledávání kontaktů ve škole, informacím o studijním oddělení, o předmětech a harmonogramu školního roku. KOSeek pro iOS má co nejvíce usnadňovat přístup k informacím ze studijního informačního systému KOS. V rámci práci jsou prozkoumány možnosti KOSapi, pomocí kterého probíhají komunikace a získávání potřebných dat z KOSu. Dále se soustředí na analýzu původní aplikace pro Android a analýzu požadavků studentů. Také bakalářská práce vysvětluje architekturu a nástroje operačního systému iOS. Na základě provedených analýz následuje návrh, implementace a testování výsledné aplikace.

Klíčová slova KOS, KOSapi, ČVUT, FIT, FEL, studijní informační systém, Swift, iOS, mobilní aplikace

Abstract

This work describes mobile application development, the KOSeek for iOS. The application is intended to help students to quickly find any information about school contacts, office for studies, subjects and academic year schedule. One of its primary goals is making access to the study information system (KOS)

data much easier. I will go through the abilities of KOSapi, the application will be communicating with in order to get access to the KOS data. This work also concerns with analysis of the application with similar goals developed for Android operating system, and requirements analysis of the CTU students. The architecture and development tools of the iOS will be described as well. Based on performed analysis, the application is designed, constructed and fully tested.

Keywords KOS, KOSapi, CTU, FIT, FEL, study information system, Swift, iOS, mobile application

Obsah

Seznam obrázků

Úvod

V naší době s větší pravděpodobností můžeme říct, že každý člověk používá chytrý mobilní telefon. Dnes pomocí kompaktního zařízení je možné rychle kontaktovat své kamarády, kolegy a získat od nich nutné informace. Proto se počet moderních přístrojů stále zvyšuje a stejně se zvyšuje počet uživatelů mobilních operačních systémů. Mezi nejpopulárnější operační systémy pro mobilní telefony patří iOS, Android a Windows Phone. Kromě základních funkcí jako vyřizování mailů a zpráv, prohlížení webu, hledání v mapách a volání uživatel iOS má možnost doplnit funkcionalitu mobilního zařízení o další software, který lze získat v App Store.

Motivací vypracování této bakalářské práce pro autora byl jeho velký zájem o programovací jazyk Swift a platformu Apple iOS, dále také absence aplikace pro iOS, která by umožnila rychlý a vhodný přístup k informacím ze studijního informačního systému KOS. KOS je webová aplikace, která slouží studentům k zapsání předmětů, tvorbě rozvrhu, zobrazení termínů zkoušek, jednorázových akcí a přihlášení na ně. Zde studenti mají své studijní výsledky a mohou kontrolovat svůj studijní plán, vkládat závěrečné práce a projevit zájem o obor.

Cíle práce

Cílem této bakalářské práce je vytvoření mobilní aplikace KOSeek pro iOS, která slouží studentům k vyhledávání kontaktů ve škole, informacím o studijním oddělení, o předmětech a harmonogramu školního roku. Aplikace má za cíl co nejvíce usnadňovat přístup k informacím ze studijního informačního systému KOS. Bude provedena analýza původní aplikace pro Android a také analýza požadavků studentů ČVUT FIT a FEL.

Struktura práce

Práce je rozdělena do následujících kapitol:

- Analýza se skládá z několika sekcí. První sekce popisuje studijní informační systém KOS. Dále následuje sekce, kde jsou popsány možnosti KOSapi a přístup k datům KOSu. Potom kapitola obsahuje analýzu původní aplikace pro Android a také analýzu požadavků studentů ČVUT FIT a FEL. Kromě toho je popsána platforma Apple iOS, její možnosti a struktura. Na závěr práce je provedena analýza funkčních a nefunkčních požadavků a také případy užití aplikace KOSeek pro iOS.
- Návrh — přehled wireframů a popis vybraných technologií pro vytvoření aplikace.
- Implementace a testování — popis jednotlivé části realizace a její testování.
- Vyhodnocení — ohodnocení aplikace, porovnání výsledné aplikace s návrhem a možnosti její rozšíření.

Kapitola 1

Analýza

Kapitola popisuje studijní informační systém KOS, možnosti KOSapi, přístup k datům z KOSu a také obsahuje analýzu existující aplikace pro Android a analýzu požadavků studentů ČVUT FIT a FEL. Dále je popsán operační systém iOS, jeho struktura a možnosti. Kromě toho kapitola zahrnuje analýzu požadavků a případy užití aplikace KOSeek pro iOS.

1.1 Studijní informační systém KOS

„*KOS slouží k podpoře studijních agend na ČVUT v Praze. Všechny funkce systému jsou realizovány jako aplikace pracující v prostředí UNIX a využívající data uložená v databázi Oracle*“ [1].

KOS — webová aplikace, která slouží studentům k zapsání předmětů, tvorbě rozvrhu, zobrazení termínů zkoušek, jednorázových akcí a přihlášení na ně. Zde studenti mají své studijní výsledky a mohou kontrolovat svůj studijní plán, vkládat závěrečné práce a projevit zájem o obor.

Webové rozhraní je rozmístěno na webové stránce http://kos.cvut.cz a nepodporuje v sobě nativní zobrazení pro mobilní zařízení s operačním systémem iOS. Avšak v iOS je možné používat KOS pomocí webového prohlížeče (například Safari). Tento způsob zobrazení informací není ideální a vhodný vzhledem k formátu webové stránky KOSu. Tehdy uživatel, aby dostal potřebné informace, má pokaždé zvětšovat a scrollovat obsah stránky.

1.2 KOSapi

Všechny potřebné informace pro aplikaci z informačního systému KOS se načítají pomocí KOSapi. „*KOSapi poskytuje aplikační rozhraní (API) v podobě RESTful webových služeb, které zprostředkovává přístup k vybrané části dat v databázi KOS. Odstraňuje nutnost zpracovávání exportů, neustálou duplikaci všech dat a potíže s jejich udržováním. RESTové služby staví na osvědčených*

konceptech webu jakožto distribuovaného prostředí vzájemně provázaných informací. KOSapi umožňuje a podporuje vznik školních i studentských aplikací, které pro svou činnost vyžadují online aplikační přístup k datům souvisejícím s výukou“ [2].

1.2.1 Representational State Transfer

REST [3] pomáhá jednoduše číst, vytvořit, editovat anebo smazat informace ze serveru pomocí jednoduchých HTTP volání. To je architektura rozhraní pro distribuované prostředí, kterou navrhl Roy Fielding v rámci své disertační práce. Všechny informace mají vlastní identifikátor URI a existují čtyři základní metody (standardní HTTP metody) pro přístup k nim:

- GET (Retrive) — získání zdroje. S touto metodou se dnes setkává každý uživatel webu — požadavek na získání obsahu stránky.
- POST (Create) — vytvoření dat. Při volání není známý identifikátor zdroje, proto se používá domluvený společný identifikátor ("endpoint").
- DELETE — smazání zdroje. Volání je obdobné volání metody GET.
- PUT (Update) — změna dat. Operace je obdobná POST volání, s tím rozdílem, že je známý konkretní URI zdroje.

Pro svou datovou výměnu REST používá tyto standardizované formáty: JSON, XML a ATOM/RSS. Webové služby využívající REST se nazývají RESTful.

1.2.2 Možnosti KOSapi

KOSapi poskytuje kromě jiných následující RESTful zdroje [4]:

- CourseEvents — jednorázové akce pro vypisování zápočtových testů, hromadných zápisů apod.
- Courses. Dvě entity předmětu: předmět (course) a tzv. instance předmětu (coursin). První entita obsahuje statické údaje předmětu — název, způsob zakončení, anotace, osnovy apod. Tyto údaje se po dobu existence předmětu nemění. Instance předmětu — konkrétní instance, která je vypsaná v semestru a obsahuje proměnné údaje jako je kapacita, počet obsazených míst apod.
- Divisions — střediska či organizační jednotky (v terminologii KOSu jde o nákladová střediska) tvoří rektorát, fakulty, katedry a tzv. podkatedry. V aplikaci je použito pro získání informací o učitelích.
- Exams — zkouškové termíny. Pomocí parametru query, lze například vyhledat termíny dle předmětu, což je využito v aplikaci.

- Semesters — semestry. Pro získání aktuálního semestru, ve kterém probíhá výuka, je možné využit zdroj current.

- Students — rozsáhlý zdroj poskytující informace o studentech, jejich studiu, zápisech předmětů a jiných.

- Teachers — zdroj vyučujících.

Všechny použité v aplikaci RESTful zdroje jsou uvedeny v 3.1.2. sekci.

1.2.3 Přístup k datům KOSapi

Rozhraní KOSapi poskytuje jenom data, která jsou veřejně dostupná z akademické obce ČVUT. Nutná autorizace pro přístup k zdrojům probíhá pomocí OAuth 2.0 protokolu, který je podrobněji popsán níže.

1.2.4 OAuth 2.0

„*OAuth 2.0 (RFC 6749) je moderní autorizační protokol (resp. framework), který se stal de facto standardem pro zabezpečení RESTových webových služeb. Jeho hlavní výhoda tkví v tom, že uživatel může poskytnout klientské aplikaci (např. Twitter klientu) přístup k jeho datům v nějaké službě (např. Twitteru), aniž by té aplikaci musel vyzradit své přístupové údaje do služby, a tím ji poskytl prakticky neomezený přístup k jeho účtu. Dále umožňuje podrobně vymezit pravomoci jednotlivých klientských aplikací (pomocí tzv. scopes) a detailně sledovat využívání poskytnutých privilegií*“ [5].

V rámci projektu KOSapi je vyvíjen autorizační server. Jedná se o samostatný OAuth 2.0, tzn. není součástí vlastní služby vystavující API (resource server), ale více samostatných služeb může využívat jeden (vzdálený) OAAS, který vydává a validuje tokeny. Server běží na adrese https://auth.fit.cvut.cz/ a může ho využívat kdokoli z akademické obce ČVUT.

Adresy jednotlivých endpointů:

- Authorization Endpoint: https://auth.fit.cvut.cz/oauth/oauth/authorize

- Token Endpoint: https://auth.fit.cvut.cz/oauth/oauth/token

Pro získání dat KOSapi požaduje tzv. access token, který v sobě obsahuje bezpečnostní informace seance a identifikuje uživatele a jeho oprávnění. Pro získání takového tokenu je potřeba vytvoření aplikaci v Apps Manageru ČVUT. Po vytvoření aplikaci je přiděleno client id a client secret (klíče nutné pro získání access tokenu) [5].

1.3 Aplikace KOSeek pro android

„Aplikace je užitečným nástrojem pro studenty i pedagogy na ČVUT. Spolupracoval jsem na ní s Bc. Adamem Šimkem pod vedením Ing. Jana Šedivého CSc. Vyhledávání kontaktů, informace o studiu, rozvrhy, detaily předmětů a paralelek či důležité informace od školy a mnohé další vlastnosti jsou v aplikaci velmi dobře zakomponované s líbivým designem“ [6].

Aplikace KOSeek pro Android má následující funkcionality:

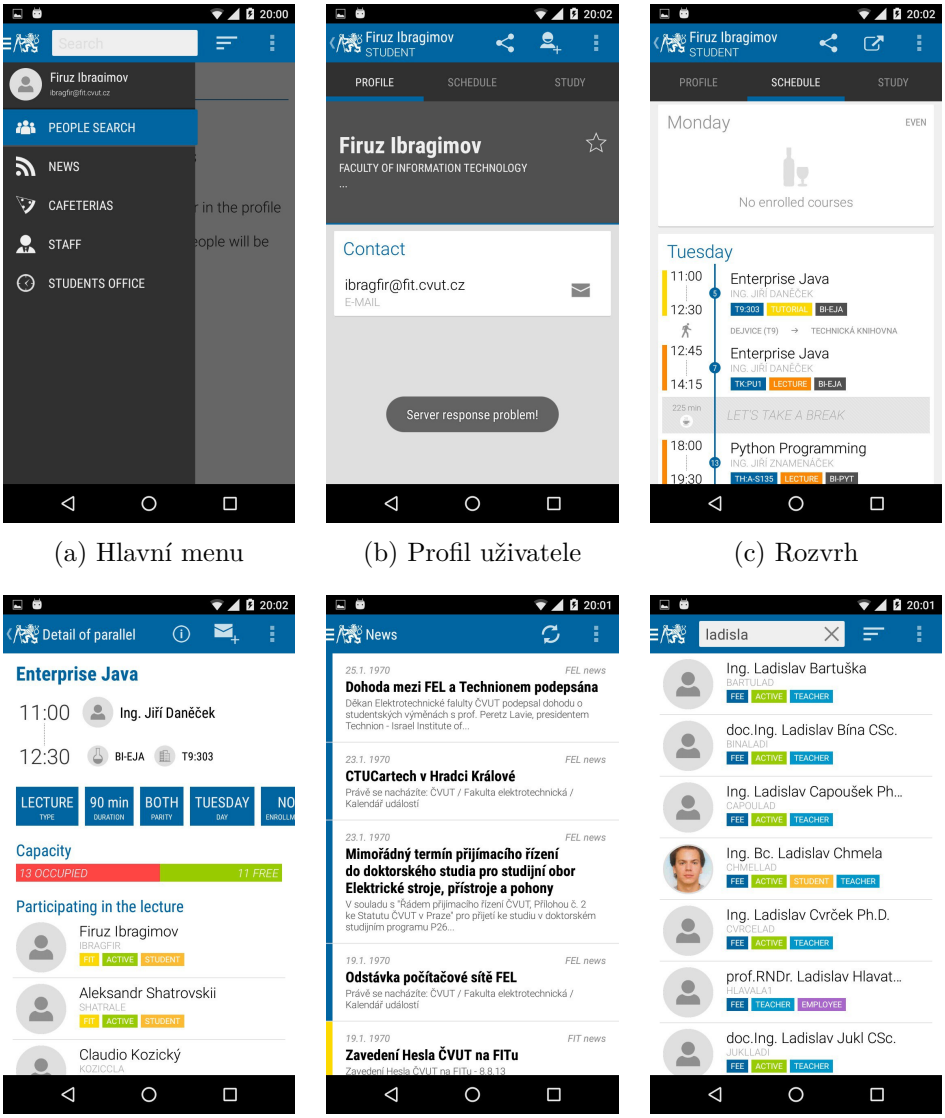
- Profil uživatele, kde jsou studijní výsledky, rozvrh uživatele a informace o studiu. Také jsou zde detaily o paralelce.
- RSS novinky, kde se zobrazují aktuality fakult FIT a FEL.
- Menu studentských jídelen.
- Výhledávání kontaktů ve škole. Zobrazí se tady kontakty na vedení, děkana, proděkany, atd. Lze rychle hledat kontakt na učitele a studenty.
- Studijní oddělení — úřední hodiny a kontakty.

Jednotlivé obrazovky lze vidět na obrázcích 1.1. KOSeek pro Android po úspěšném přihlášení stahuje všechna potřebná data do databáze, které pak lze využít v offline režimu. Aplikace funguje celkem dobře, ale občas se nastávají problémy, nejspíš se serverem KOSapi, a aplikace padá.

1.4 Analýza aplikace pro jiný informační systém

Pro tyto účely je vybraná webová aplikace SIS pro informační systém Karlovy univerzity (UK). *„SIS je určen zaměstnancům a studentům vysokých škol a dále pak zájemcům o studium na těchto školách. V závislosti na postavení uživatele vzhledem k dané škole jsou jednotlivým uživatelům přiřazena práva*“ [7]. Hlavní stránka obsahuje několik sekcí: výuka, rozvrh, přijímací řízení a jiné (viz obrázek 1.2).

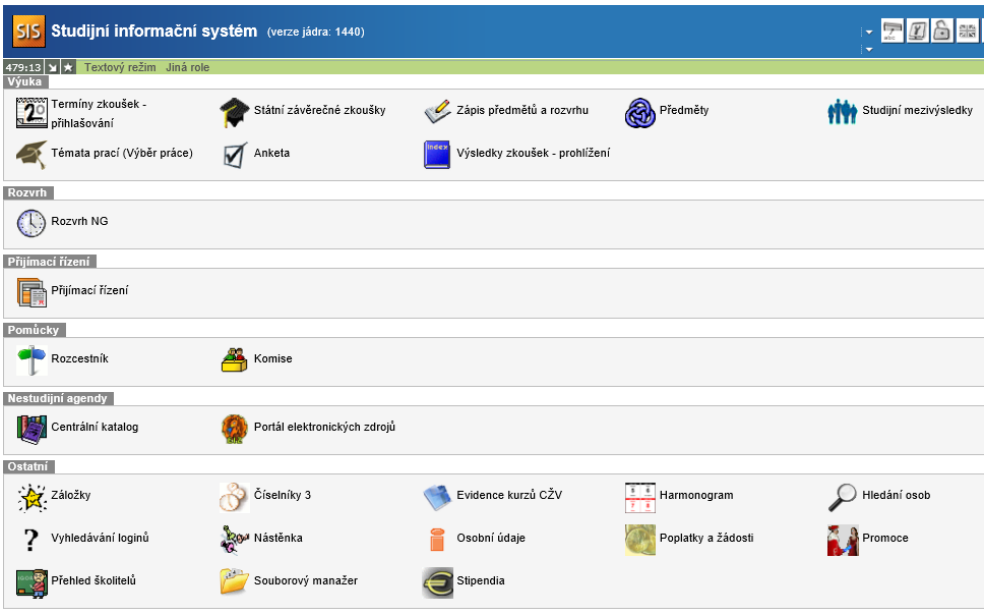
Na stránce zápis předmětů a rozvrhu jsou dvě možnosti zápisu předmětů: zápis studijní plán, kde je možné zapisovat pouze předměty uvedené v plánu a zápis vlastní, kde lze zapisovat libovolný předmět na UK (příslušná fakulta musí mít povolen zápis předmětů). Stránka zapsané zobrazuje všechny zapsané studentem předměty ve vybraném semestru. Zde jsou informace o počtu kreditů, rozsahu výuky, fakultě, typu předmětů a také jsou zobrazeny rozvrhové paralelky (viz obrázek 1.3). Při kliknutí na předmět se zobrazí detail předmětu, kde jsou uvedené následující informace o předmětu: anotace, cíl, literatura, fakulta, semestr, kredity, jazyk a způsob výuky (viz obrázek 1.4). Rozvrh lze zobrazit dle učeben, předmětů, budov a také je možnost zobrazit rozvrh učitelů a rozvrh pouze těch studentů, kteří nemají skryté osobní údaje, což každý student může provést v systému (viz obrázek 1.5).



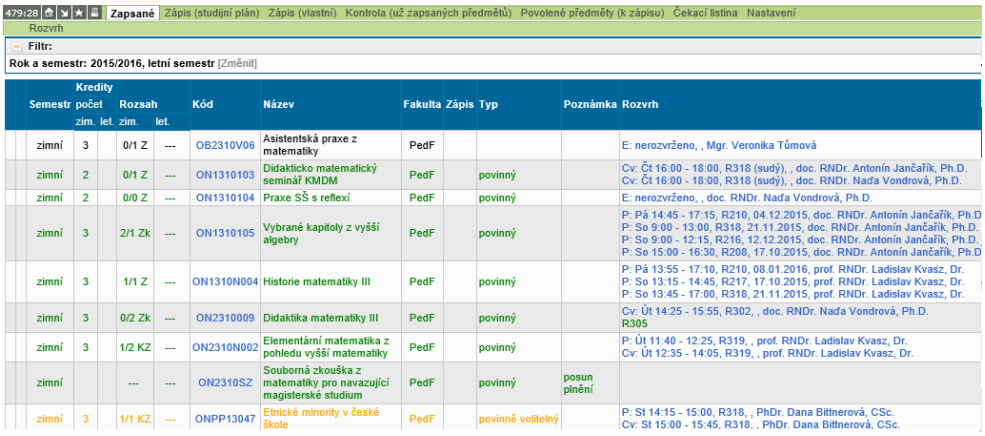
(a) Hlavní menu (b) Profil uživatele (c) Rozvrh

(d) Detaily paralelky (e) Novinky (f) Vyhledávání kontaktů

Obrázek 1.1: KOSeek pro Android

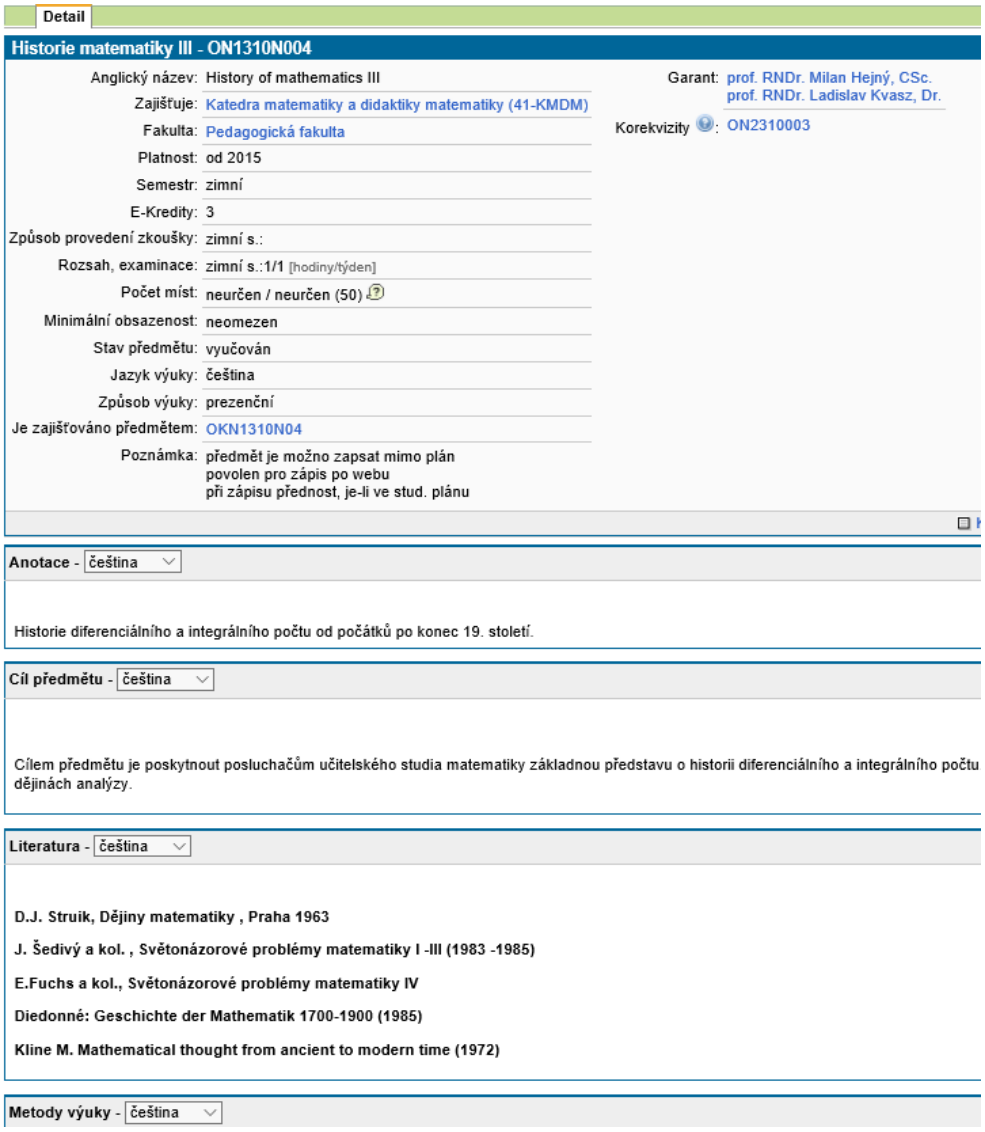


Obrázek 1.2: Informační systém SIS

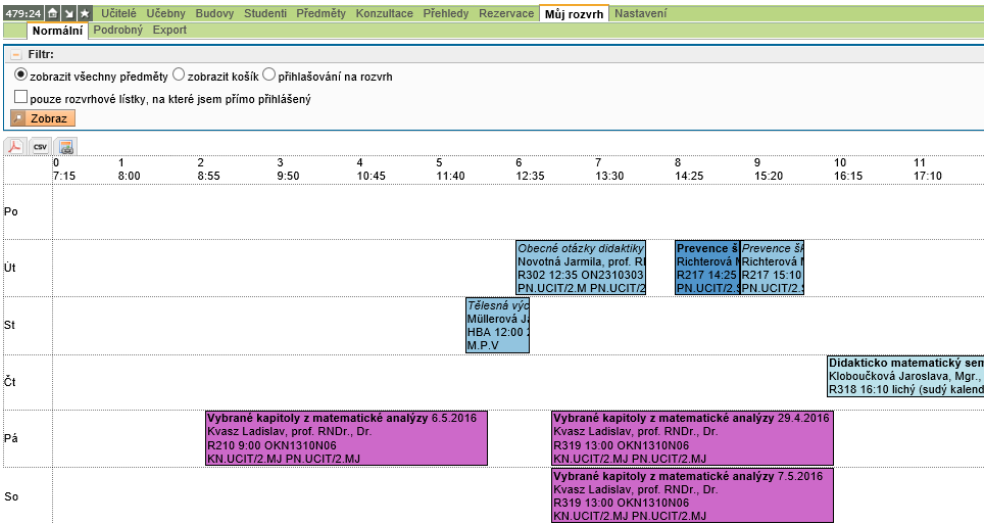


Semestr	Kredity počet zim.	Kredity počet let.	Rozsah zim.	Rozsah let.	Kód	Název	Fakulta	Zápis	Typ	Poznámka	Rozvrh
zimní	3		0/1 Z	---	OB2310V06	Asistentská praxe z matematiky	PedF				E: nerozvrženo, , Mgr. Veronika Tůmová
zimní	2		0/1 Z	---	ON1310103	Didakticko matematický seminář KMDM	PedF		povinný		Cv: Čt 16:00 - 18:00, R318 (sudý), , doc. RNDr. Antonín Jančařík, Ph.D. Cv: Čt 16:00 - 18:00, R318 (sudý), , doc. RNDr. Naďa Vondrová, Ph.D.
zimní	2		0/0 Z	---	ON1310104	Praxe SŠ s reflexí	PedF		povinný		E: nerozvrženo, , doc. RNDr. Naďa Vondrová, Ph.D.
zimní	3		2/1 Zk	---	ON1310105	Vybrané kapitoly z vyšší algebry	PedF		povinný		P: Pá 14:45 - 17:15, R210, 04.12.2015, doc. RNDr. Antonín Jančařík, Ph.D. P: So 9:00 - 13:00, R318, 21.11.2015, doc. RNDr. Antonín Jančařík, Ph.D. P: So 9:00 - 12:15, R216, 12.12.2015, doc. RNDr. Antonín Jančařík, Ph.D. P: So 15:00 - 16:30, R208, 17.10.2015, doc. RNDr. Antonín Jančařík, Ph.D.
zimní	3		1/1 Z	---	ON1310N004	Historie matematiky III	PedF		povinný		P: Pá 13:55 - 17:10, R210, 08.01.2016, prof. RNDr. Ladislav Kvasz, Dr. P: So 13:15 - 14:45, R217, 17.10.2015, prof. RNDr. Ladislav Kvasz, Dr. P: So 13:45 - 17:00, R318, 21.11.2015, prof. RNDr. Ladislav Kvasz, Dr.
zimní	3		0/2 Zk	---	ON2310009	Didaktika matematiky III	PedF		povinný		Cv: Út 14:25 - 15:55, R302, , doc. RNDr. Naďa Vondrová, Ph.D. R305
zimní	3		1/2 KZ	---	ON2310N002	Elementární matematika z pohledu vyšší matematiky	PedF		povinný		P: Út 11:40 - 12:25, R319, , prof. RNDr. Ladislav Kvasz, Dr. Cv: Út 12:35 - 14:05, R319, , prof. RNDr. Ladislav Kvasz, Dr.
zimní			---	---	ON2310SZ	Souborná zkouška z matematiky pro navazující magisterské studium	PedF		povinný	posun plnění	
zimní	3		1/1 KZ	---	ONPP13047	Etnické minority v české škole	PedF		povinně volitelný		P: St 14:15 - 15:00, R318, , PhDr. Dana Bittnerová, CSc. Cv: St 15:00 - 15:45, R318, , PhDr. Dana Bittnerová, CSc.

Obrázek 1.3: SIS — zapsané předměty



Obrázek 1.4: SIS — detail předmětu



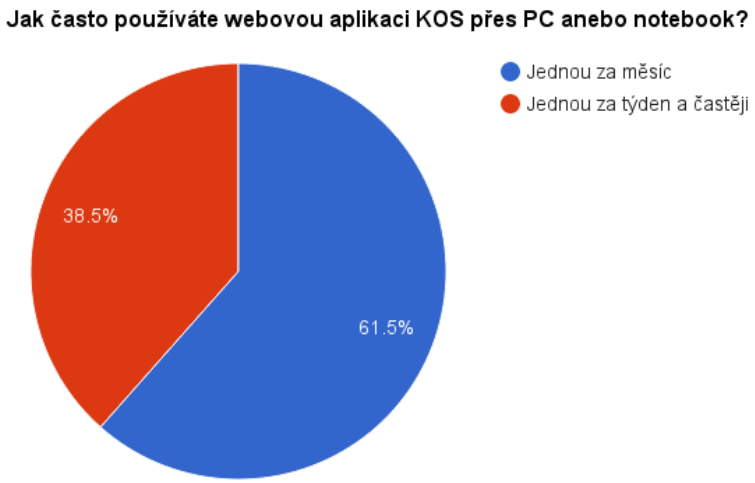
Obrázek 1.5: SIS — rozvrh studenta

1.5 Analýza požadavků studentů

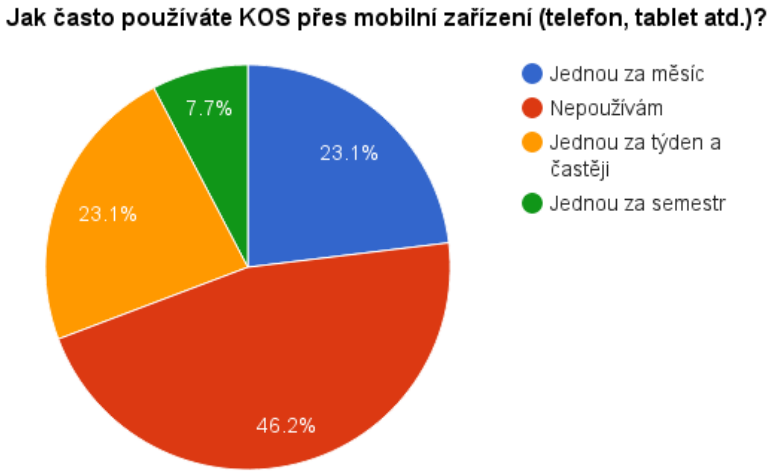
Analýza požadavků studentů je provedena na základě vytvořeného a odeslaného studentům dotazníku, který obsahuje následující otázky:

1. Jak často používáte webovou aplikaci KOS přes PC anebo notebook?
2. Jak často používáte KOS přes mobilní zařízení (telefon, tablet atd.)?
3. Chtěl(a) byste mít aplikaci KOSeek pro iOS?
4. Jaké funkce KOSu nejčastěji používáte?
5. Jaké funkce byste chtěl(a) mít v aplikaci KOSeek pro iOS?

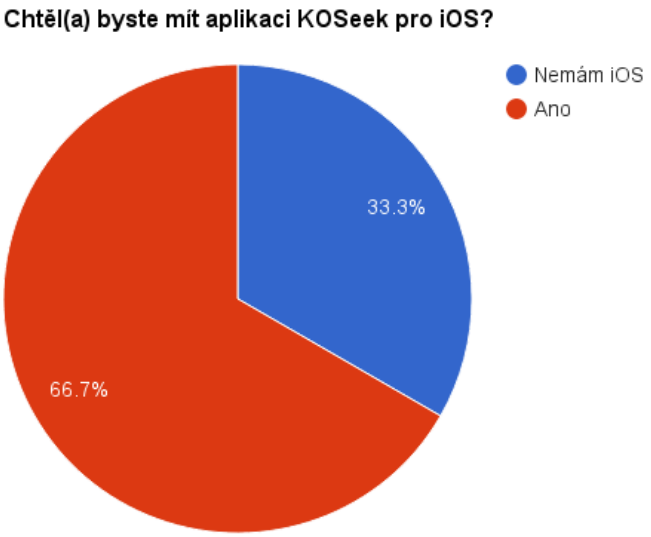
Diagram 1.6 ukazuje odpovědi studentů na první otázku. Podle diagramu lze vidět, že 61,5 % studentů používají webovou aplikaci KOS jednou za měsíc, ostatní 38,5 % používají KOS jednou za měsíc anebo častěji. Většinou studenti nepoužívají KOS přes mobilní zařízení (viz diagram 1.7). Důvodem k tomu může být absence aplikace, která by umožnila rychlý a vhodný přístup k informacím ze studijního informačního systému KOS. Což potvrzuje diagram 1.8, dle kterého 66,7 % studentů by chtěli mít aplikaci KOS pro iOS. Diagram 1.9 ukazuje v procentech nejpoužívanější funkce KOSu dle odpovědí studentů. Podle poslední otázky je zjištěno, jaké funkce chtějí mít studenti a také podle odpovědí je vytvořen diagram 1.10. Ostatní odpověď je pouze jedna — rozvrh přátel. Tato funkce není přidána do aplikace kvůli nedostatkům prav vidět rozvrhy studentů.



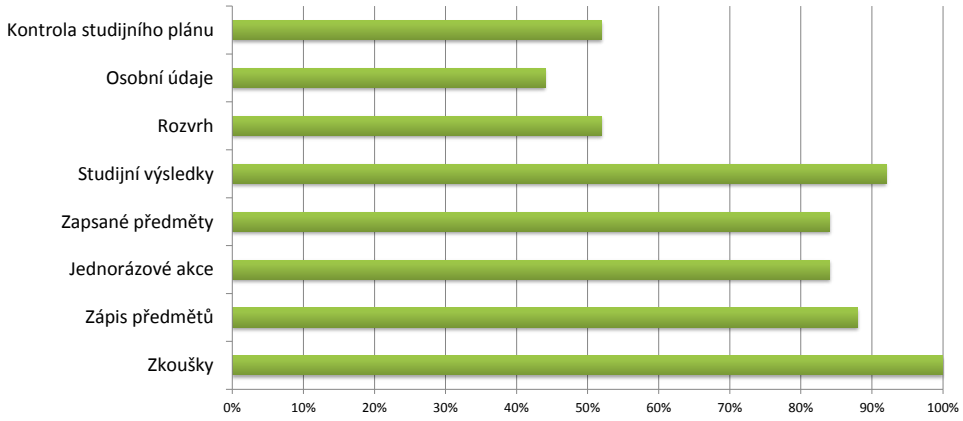
Obrázek 1.6: Diagram používání webové aplikace KOS studenty



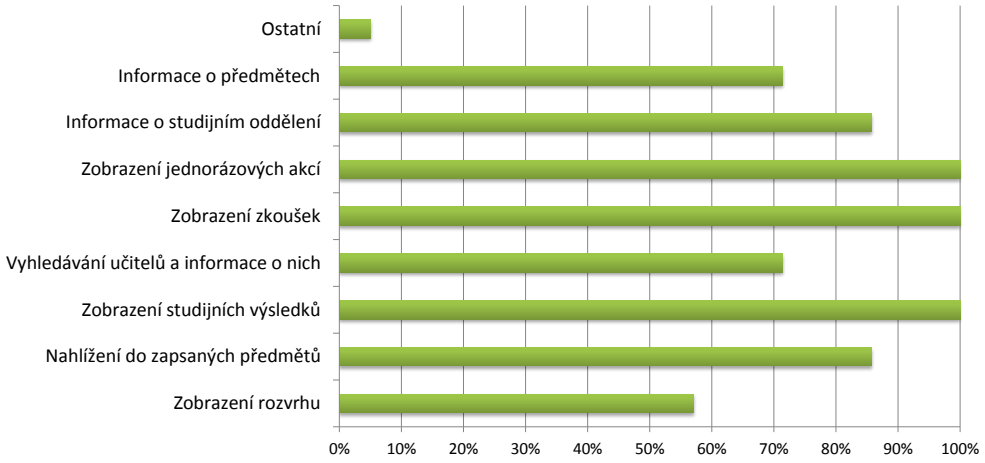
Obrázek 1.7: Diagram používání KOS přes mobilní zařízení



Obrázek 1.8: Diagram potřebnosti aplikace pro iOS dle názoru studentů



Obrázek 1.9: Diagram nejpoužívanějších funkcí KOSu



Obrázek 1.10: Diagram funkcí aplikace, které chtějí mít studenti

1.6 Apple iOS

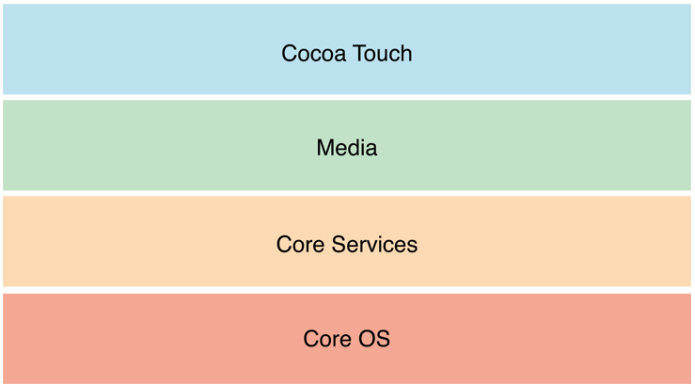
iPhone OS je mobilní operační systém UNIXového typu založený na jádru Hybrid (XNU), který je vytvořen společností Apple Inc. Operační systém je primárně určen pro mobilní zařízení, ale později se začal používat na jiných zařízení Apple — iPad, iPod a Apple TV. iOS spravuje hardware zařízení a poskytuje technologie nezbytné pro implementaci nativních aplikací. Také iOS nabízí různé systémové aplikace, jako jsou Telefonování, Maily a prohlížeč Safari, které představují standardní služby pro klienta.

Pro vývojáře Apple poskytuje iOS SDK, nástroje a rozhraní, které jsou nutné k vývoji, instalaci, spuštění a testování nativních aplikací vyskytujících na hlavní obrazovce v iOS. Architektura operačního systému iOS je představená jako množina vrstev (viz obr. 1.11). Nižší vrstvy obsahují základní služby a technologie. Vyšší vrstvy jsou založeny na nižších a poskytují více sofistikované rozhraní. „*The higher-level frameworks are there to provide object-oriented abstractions for lower-level constructs. These abstractions generally make it much easier to write code because they reduce the amount of code you have to write and encapsulate potentially complex features, such as sockets and threads*“ [8].

1.6.1 Vrstva Core OS

Nejnižší vrstva, která obvykle při vývoji není přímo použita, ale určitě je využita vyššími frameworky. Pokud je nutné explicitně pracovat s bezpečností anebo komunikovat s vnějším hardware příslušenstvím, tak tuto vrstvu je potřeba využít přímo. Core OS poskytuje následující rozhraní: [9]

- Accelerate Framework — rozhraní obsahující nástroje pro matematický



Obrázek 1.11: Vrstvy iOS

výpočet matic, vektorů. Také framework slouží k zpracování digitálních signálů a manipulaci s velkými čísly.

- Core Bluetooth Framework — povoluje vývojářům používat Bluetooth low energy pro komunikaci s zařízeními. Rozhraní poskytuje funkce pro hledání, připojení a odpojení.
- External Accessory Framework — podporuje komunikaci s hardware přislušenstvím, která jsou připojena k zařízení s iOS.
- Generic Security Services Framework — poskytuje standardní množinu bezpečných služeb iOS aplikacím.
- Local Authentication Framework — použití Touch ID pro autentizaci uživatele.
- Network Extension Framework — podpora konfigurace a správy VPN tunelů.
- Security Framework — garance bezpečnosti dat, která používá aplikace.

1.6.1.1 Systém

„*The system level encompasses the kernel environment, drivers, and low-level UNIX interfaces of the operating system. The kernel itself, based on Mach, is responsible for every aspect of the operating system. It manages the virtual memory system, threads, file system, network, and interprocess communication*“ [9]. iOS poskytuje množinu rozhraní pro přístup k nízkoúrovňovým funkcím operačního systému. Tyto funkce jsou přístupné přes knihovnu LibSystem, která je založena na C, a podporují následující:

- Concurrency (POSIX vlákna a Grand Central Dispatch)

- Síťování (BSD sockety)
- Přístup k souborovými systému
- Standardní vstup/výstup
- Bonjour a DNS služby
- Lokální informace
- Alokace paměti
- Matematické výpočty

1.6.2 Core Services

Nízkoúrovňová vrstva, která obsahuje zásadní systémové služby. Hlavními frameworky jsou Core Foundation a Foundation definující základní typy, které používají všechny aplikace. Také Core Services podporuje technologie jako lokace, iCloud, sociální média a síťování. Klíčové technologie: [10]

- Peer-to-Peer Services — peer-to-peer spojení přes Bluetooth, které se primářně používá ve hrách.
- iCloud Storage — povoluje aplikacím zapisovat uživatelská data, která jsou potom dostupná z počítače a iOS zařízení.
- Data Protection — práce s důležitými daty, která jsou ukládána ve šifrovaném formátu. Pokud je zařízení zablokováno, tak chráněná data nejsou dostupná ani aplikaci, ani potenciálnímu hackerovi. Ovšem když zařízení není zablokováno, tak se vytváří dešifrovací klíč pro přístup aplikací k datům.
- Grand Central Dispatch — BSD technologie pro správu plnění úkolů. GCD kombinuje asynchronní programovací model s vysoce optimalizovaným jádrem s účelem poskytnout alternativu threading procesu.
- SQLite — zjednodušená verze SQL databáze bez nutnosti spuštění samostatného serverového procesu.
- XML Support — poskytuje třídu NSXMLParser pro získání elementů z XML souboru a práci s nimi.

Core Services poskytuje následující rozhraníi:

- Address Book Framework — programovací přístup k uživatelské databázi kontaktů.
- CFNetwork Framework — množina efektivních C funkcí pro práci se síťovými protokoly.

- CloudKit Framework — poskytuje kanál pro výměnu dat mezi aplikací a iCloud.
- Core Data Framework — technologie pro řízení datových modelů technologie model-view-controller (viz 1.6.5) aplikace. Místo programování vlastních datových struktur, je lepší využit grafické nástroje v XCode IDE pro vytváření schémat představujících datové modely. V runtime instance datových entit jsou vytvářeny, řízeny a přístupné přes Core Data framework. Rozhraní spravuje datové modely a tím zmenšuje počet řádků kódu. Core Data poskytuje následující funkcionality:
  - Ukládání objektových dat v SQLite databázi pro optimální výkon.
  - Třída NSFetchedResultsController pro správu řízených výsledků, které se mají zobrazit v tabulkovém view.
  - Správa undo/redo kroků.
  - Podpora validace hodnot vlastností.
  - Zajištění konzistence vztahů mezi objekty při změnách.
  - Podpora seskupení, filtrování a organizace dat v paměti.
- Core Foundation Framework — množina efektivních C funkcí poskytující zásadní služby iOS aplikacím. Framework podporuje:
  - Kolekce (pole, množiny atd.)
  - Bundles
  - Správa řetězců
  - Správa dat a času
  - Správa nastavení
  - Manipulace s potoky a URL
  - Vlákna
  - Porty a sockety
- Foundation Framework — poskytuje třídu NSXMLParser pro získání elementů z XML souboru a práci s nimi.

1.6.3 Vrstva Media

Media vrstva poskytuje aplikacím grafické, audio a video technologie: [11]

- Grafické technologie — pomáhají přidávat uživatelský design a grafiku do obrazovky:
  - UIKit graphics — podpora kreslení obrázků a Beziérových křivek a animace obsahu view. Technologie poskytuje rychlou a efektivní cestu renderování obrazu a textového obsahu.

  - Core Graphics framework — nativní engine kreslení pro iOS aplikace poskytující uživatelské 2D vektorové a obrazové renderování. Rozhraní není tak rychlé jako je OpenGL ES, ale je velmi vhodné pro situace, kdy je potřeba dynamické renderování vlastních 2D forem a obrazu.
  - Core Animation — technologie pro práci s animací.
  - OpenGL ES a GLKit — OpenGL ES spravuje moderní 2D a 3D renderování s použitím rychlých hardware rozhraní. Obvykle se používá vývojáři her anebo tím, kdo chce realizovat uchvacující grafický zážitek. GLKit je množina Objective-C tříd poskytující možnosti OpenGL ES jako objektově orientované rozhraní.
  - Image I/O — rozhraní pro čtění a zápis nejznámějších formátů obrázků.

- Audio technologie — práce s hardware s cílem poskytnout kvalitní zvuk uživatelům:
  - Media Player framework — přístup k uživatelově knihovně iTunes a podpora přehrávání playlistů a tracků.
  - AV Foundation — Objective-C rozhraní pro správu natáčení a přehrávání audií a videí.
  - OpenAL — standardní technologie poskytnutí polohového zvuku.
  - Core Audio — množina frameworků poskytující obyčejné a sofistikované rozhraní pro zápis a přehrávání audio a MIDI obsahu.

- Video technologie — podpora řízení statického videa anebo přehrávání streaming obsahu z internetu:
  - UIImagePickerController — třída UIKit view controlleru pro vybrání uživatelských média souborů.
  - AVKit — množina lehce použitelných rozhraní pro prezentování videí. Podporuje full-screen a partial-screen přehrávání.
  - AV Foundation — moderní přehrávání videí a možnost natáčení.
  - Core Media — definuje nízkoúrovňové typy dat a rozhraní pro manipulaci s médii.

1.6.4 Cocoa Touch

Vrstva obsahující nejdůležitější rozhraní pro tvorbu iOS aplikací. Tyto rozhraní definují jak bude výsledná aplikace vypadat. Kromě toho rozhraní poskytují základní strukturu aplikace a podporují klíčové technologie jako jsou multitasking, dotykový vstup, notifikace a další vysokoúrovňové systémové služby. Klíčové technologie: [12]

- App Extensions — rozhraní umožňující rozšířit nějaké funkcionality. Například je možné vytvořit rozšíření poskytující vlastní klávesnici, kterou si uživatelé mohou nastavit místo systémové.
- TextKit — množina tříd pro spravování textů a typografie. Tato technologie poskytuje hodně způsobu zobrazení textu, který pak může editovat uživatel.
- UIKit Dynamics — přidání dynamických vlastností GUI prvkům jako jsou kolize, gravitace a vektory síly.
- Multitasking — model vytvořený s účelem prodloužení životnosti baterie a zároveň dává aplikacím čas, potřebný pro kritickou práci.
- Auto Layout — pomocí této technologie je možné definovat pravidla, které popisuje jak se mají rozmístit GUI prvky.
- Storyboards — doporučený způsob tvorby designu iOS aplikace. Dává k dispozici možnost vytvářet interface na jednom místě, kde jsou vidět všechny view a view controllery a pochopit jak spolu pracují.
- Gesture Recognizers — rozpoznání různých typů gest, jako jsou swipe a pinch. Gesta jde přidat do nějakého view a spojit s vlastní metodou, která se má zavolat při detekci těchto gest.
- Apple Push Notification Service — cesta jak upozornit uživatele o novinky i když aplikace není spuštěna.
- Standard System View Controllers — systémové frameworky definující view controllery pro standardní rozhraní.

Cocoa Touch kromě jiných obsahuje následující rozhraní:

- Address Book UI Framework — Objective-C framework zobrazující standardní systémové rozhraní pro vytváření nových kontaktů, editaci a vybrání existujících kontaktů.
- EventKit UI Framework — poskytuje view controllery zobrazující standardní systémové rozhraní pro prohlížení a editaci kalendářových akcí.
- GameKit Framework — implementuje podporu Game Centru, který povoluje uživatelům sdílet jejich herní informace online.
- MapKit Framework — scrolovatelná mapa, u které lze upravit vzhled a obsah.
- Message UI Framework — podpora vytváření emailů anebo SMS zpráv.

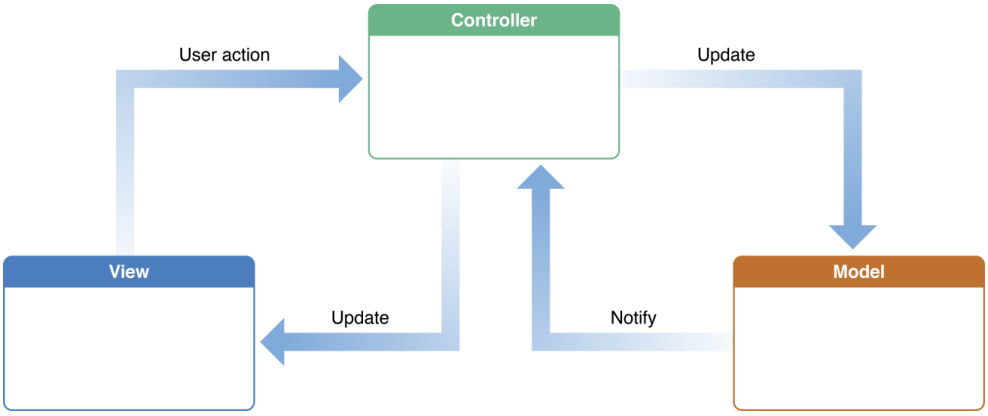
- UIKit Framework — zásadní infrastruktura pro implementaci grafických, event-driven aplikací, zahrnující následující:
  - Základní řízení aplikace a infrastruktury včetně hlavního cyklu aplikace
  - Správa UI včetně podpory storyboardů a nib souborů
  - View controller model pro zapouzdření obsahu UI
  - Objekty reprezentující standardní systémové view a ovladače
  - Podpora řízení dotykových a pohybových událostí
  - Podpora souborového modelu zahrnujícího integraci s iCloud
  - Podpora multitaskingu
  - Podpora tisku
  - Možnost úpravy vzhledu standardních UIKit prvků
  - Podpora animace UI obsahu
  - Vytváření pdf souborů

1.6.5 Model View Controller

Rozhraní iOS používají tzv. model-view-controller (MVC) návrhový vzor, který rozděluje datový model aplikace, uživatelská rozhraní a řídicí logiku do tři nezávislých komponent tak, aby změny některé z těchto komponent měly minimální vliv na ostatní (viz obr. 1.12). Tento návrh definuje nejenom komponenty, ale také i způsob jak mezi sebou komunikují. Datový model (model objects) zapouzdří specifická data aplikace a definuje logiku manipulace s těmito daty. Uživatelské rozhraní (view objects) jsou objekty, které může vidět uživatel aplikace. Tyto objekty obsahují informace jak se mají vykreslovat a jak reagovat na akce uživatele. Řídicí logika (controller objects) působí jako prostředník mezi jedním nebo více view objekty a jedním nebo více modelovými objekty. Podle obrázku 1.12 lze vidět, že akce uživatele jako vytvoření anebo modifikace dat ve vrstvě view přes řídicí logiku ovlivňuje modelové objekty. Pokud se změní modelový objekt, tak se vyšle notifikace řídicí logice, která pak obnovuje view objekty [13].

1.6.6 Struktura iOS aplikace

Vstupním bodem každé aplikace, která je založena na C, je main funkce a aplikace iOS se nijak neliší. Rozdíl je v tom, že vývojář nemusí psát main funkci sám. Místo toho XCode IDE vytvoří funkci za něho jako základní část projektu. Kód main funkce:



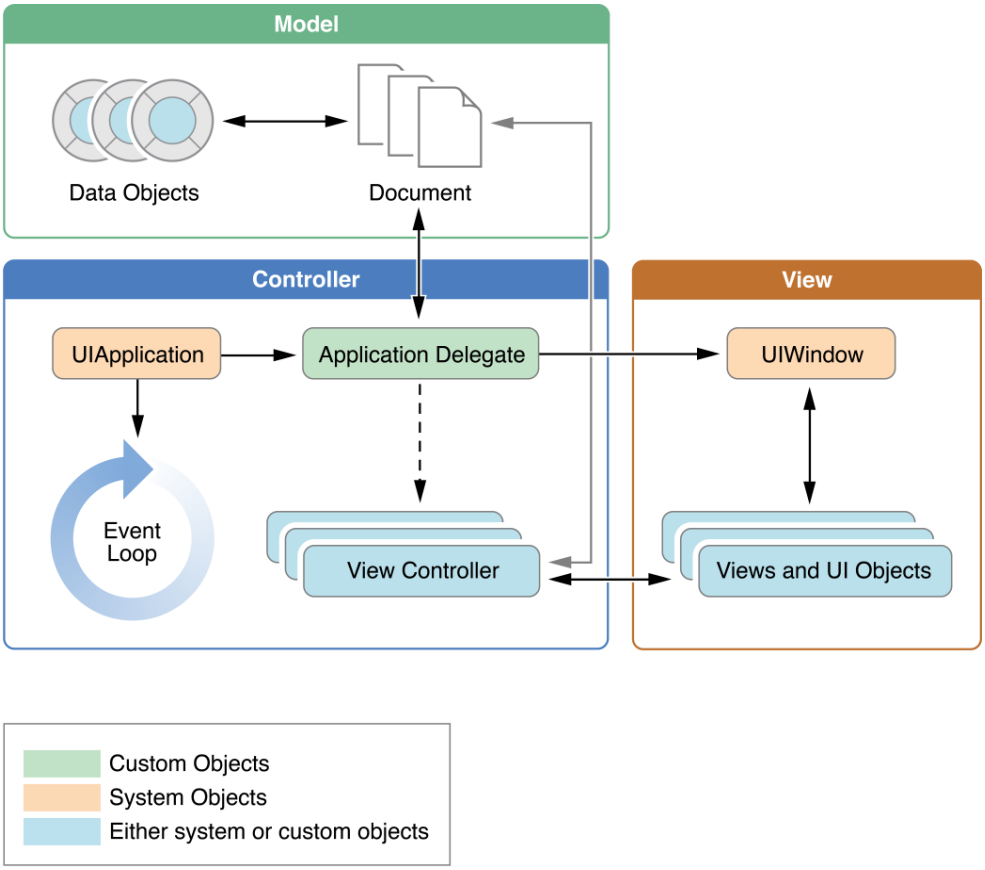
Obrázek 1.12: Model view controller návrhový vzor

```
#import <UIKit/UIKit.h>
#import "AppDelegate.h"

int main(int argc, char * argv[])
{
    @autoreleasepool {
        return UIApplicationMain(argc, argv, nil, NSStringFromClass([
                    AppDelegate class]));
    }
}
```

Úkolem této funkce je předání řízení UIKit frameworku. Funkce UIApplicationMain řídí proces vytvoření core objektů aplikace, vytváří UI z dostupných storyboard souborů, volá vlastní kód, který povoluje inicializovat nastavení a spouští hlavní cyklus aplikace. Funkce UIApplicationMain při inicializaci nastavuje klíčové objekty a pouští aplikaci. Hlavním objektem celé iOS aplikace je UIApplication objekt, který usnadňuje interakci mezi systémem a jinými objekty v aplikaci.

Na obrázku 1.13 jsou představeny běžné objekty iOS aplikací. Application delegát pracuje s UIApplication objektem a spravuje inicializaci aplikace, stavové přechody a vysokoúrovňové aplikační akce. Tento objekt je pouze jeden pro celou aplikaci a často se používá pro výchozí nastavování datových struktur. View controllery spravují prezentaci obsahu aplikace. Jeden view controller řídí jedno view a všechny jeho subview. Základní třídou pro všechny view controllery je třída UIViewController, která poskytuje výchozí funkce pro inicializaci view, prezentaci, otáčení po příslušném otáčení zařízení a několik dalších běžných systémových chování. UIWindow objekt koordinuje prezentaci jednoho anebo více view na obrazovce. Většina aplikací mají pouze jedno okno, které představuje obsah na hlavní obrazovce, ale aplikace mohou mít doplňující okno pro zobrazení obsahu na vnějším displeji. Při změně obsahu

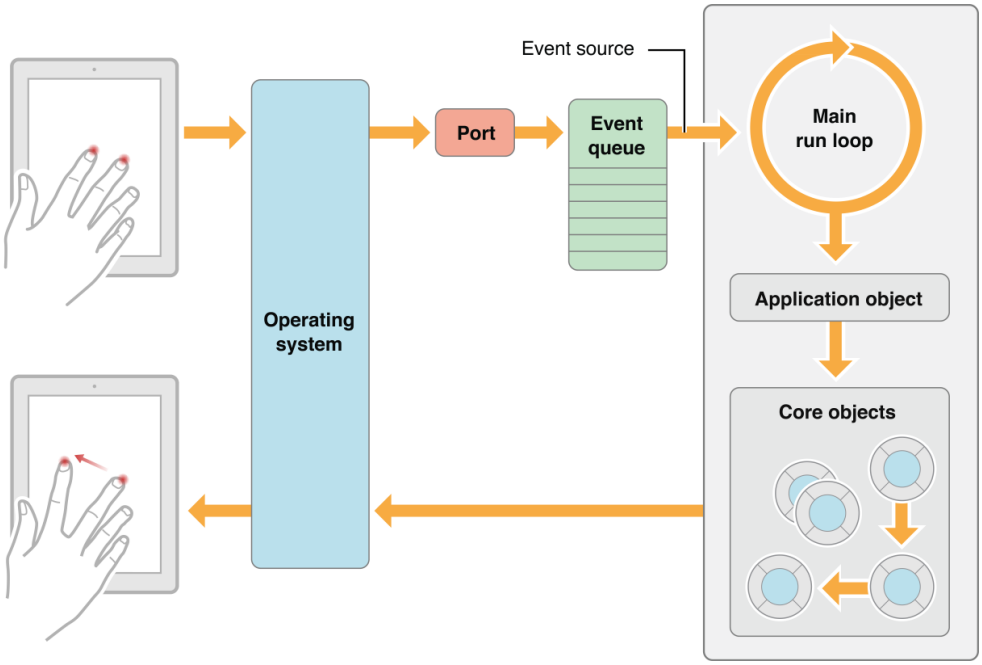


Obrázek 1.13: Běžné objekty iOS aplikací

aplikace se používá view controller pro změnu zobrazených view v okně, ale nikdy se mění okno [13].

1.6.7 Hlavní cyklus běhu

Hlavní cyklus běhu aplikace (the main run loop) zpracovává všechny související s uživatelem události. UIApplication objekt nastavuje hlavní cyklus běhu za dobu spuštění aplikace a používá ho pro zpracování události a změn view. Cyklus se provádí v hlavním vlákně aplikace, což zajišt'uje to, že uživatelské události jsou zpracovány v pořadí, ve kterém byly obdrženy. Obrázek 1.14 znázorňuje architekturu hlavního cyklu běhu a také jak operační systém zpracovává uživatelské události. Jakmile uživatel interaguje se zařízením, tak se příslušné události těchto interakcí generují systémem a dostávají se k aplikaci přes speciální port naznačený rozhraním UIKit. Události čekají ve frontě a zpracovávají se po jedné v hlavním cyklu běhu [13].



Obrázek 1.14: Zpracování událostí v hlavním cyklu běhu

1.6.8 Stavy aplikace

V libovolném okamžiku aplikace může být v jednom z následujících stavů:

- Nespuštěný — aplikace nebyla spuštěna anebo byla ukončena systémem.
- Neaktivní — aplikace je spuštěna na popředí, ale v daném okamžiku nezpracovává žádné události.
- Aktivní — aplikace běží na popředí a zpracovává události.
- Na pozadí — aplikace běží na pozadí a vykonává kód. Většina aplikací vstupuje do tohoto stavu po krátkou dobu před zastavením.
- Pozastavený — aplikace běží na pozadí, ale nevykonává kód. Systém automaticky převádí aplikaci do tohoto stavu a neposílá ji žádné notifikace před touto operací.

Na obrázku 1.15 jsou ukázány výše popsané stavy a přechody mezi nimi. Většina přechodů mezi stavy je doprovázena odpovídajícím voláními metod delegáta aplikace. Tyto metody je možností vývojáře reagovat na změny stavů vhodným způsobem: [13]

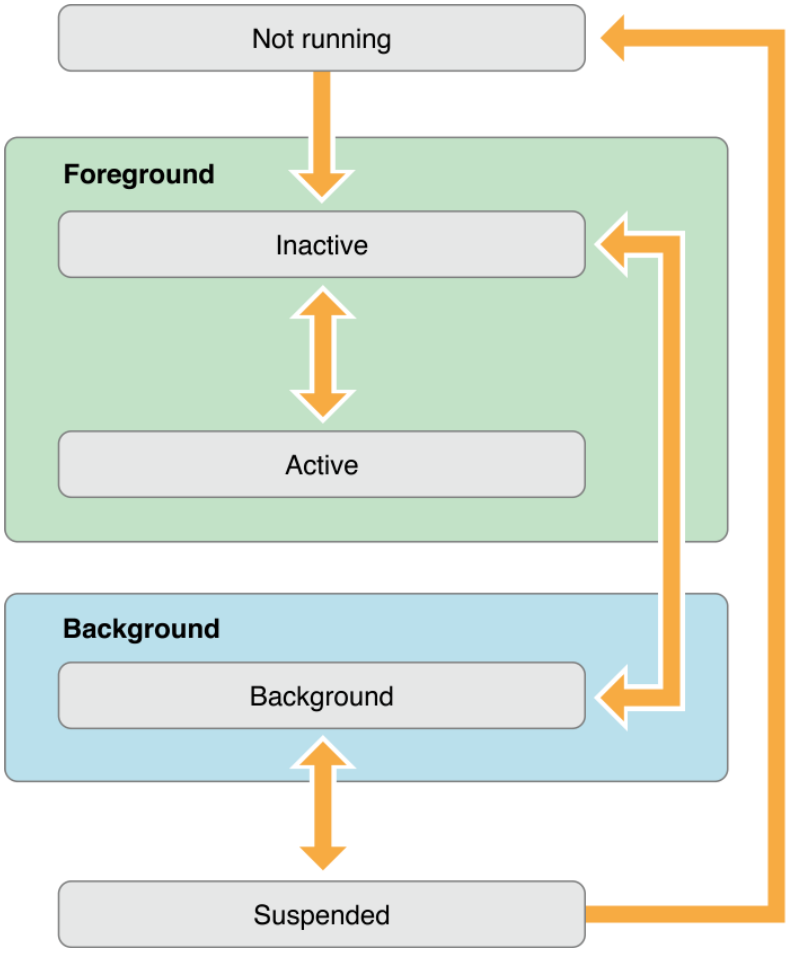
- willFinishLaunchingWithOptions — metoda je první šancí vývojáře vykonat vlastní kód za dobu běhu.

- didFinishLaunchingWithOptions — tato metoda povoluje provést finální inicializaci před tím než aplikace je zobrazena uživateli.
- applicationDidBecomeActive — informuje aplikaci o přechodu do aktivního stavu. Používá se pro poslední připravy.
- applicationWillResignActive — informuje aplikaci o přechodu z aktivního stavu. Používá se pro přechod aplikace do klidového stavu.
- applicationDidEnterBackground — metoda informuje, že teď aplikace běží na pozadí a může být kdykoliv pozastavena.
- applicationWillEnterForeground — volá se při přechodu aplikace ze stavu na pozadí do stavu na popředí, ale aplikace není aktivní.
- applicationWillTerminate — informuje aplikaci o její ukončení. Tato metoda se nevolá, pokud je aplikace pozastavena.

1.7 Analýza požadavků

1.7.1 Funkční požadavky

- Zobrazení rozvrhu uživatele. Zobrazí se všechny zapsané rozvrhové výuky dle vybrané parity.
- Nahlížení do zapsaných předmětů v aktuálním semestru. U každého předmětu je uveden: název, kód, počet kreditů a ukáže se, zda-li je předmět ukončen.
- Zobrazení studijních výsledků v jednotlivých semestrech. Uživatel si zvolí semestr podle kterého je vypsán seznam předmětů, počet získaných a zapsaných kreditů v semestru. Také počty kreditů všech semestrů.
- Vyhledávání učitelů.
- Zobrazení zkoušek podle předmětu. Uživatel si zvolí předmět podle kterého se má vrátit seznam aktuálních vypsaných termínů v KOSu. Každý záznam obsahuje datum, čas, místo, počet obsazených míst, kapacitu termínu a nejpozdější termín odhlášení.
- Zobrazení jednorázových akcí podle předmětu. Stejná funkcionalita jako u zkoušek.
- Obnovení všech dat manuálně uživatelem.
- Informace o studijním oddělení. Zobrazení úředních hodin ze studijního oddělení.



Obrázek 1.15: Stavy iOS aplikací a jejich přechody

- Při kliknutí na předmět, bud' v semestru nebo ve studijních výsledcích, objeví se obrazovka s detaily předmětu, kde jsou anotace, osnova přednášek, osnova cvičení a jiné detaily předmětu.

Po otevření aplikace uživatel se může přihlásit pomocí svých údajů stejně jako do systému ČVUT. Dále při úspěšném přihlášení se všechna data kromě zkoušek a detailů předmětů stáhnou a následně se uloží do databáze. Zkoušky se stáhnou při vybrání předmětu. Uživatel má možnost obnovit všechna data anebo jednotlivou část dat pomocí známé funkce pull to refresh v příslušné obrazovce.

1.7.2 Nefunkční požadavky

- Intuitivní grafické uživatelské rozhraní.

- Dostupnost dříve stažených informací v režimu offline.
- Stahování dat ve dvou jazycích: český a anglický.
- Lokalizace aplikace do jazyků: český, ruský (původní jazyk — anglický).

Dalším nefunkčním požadavkem je landscape orientace obrazovek. Ačkoliv v poslední době aplikace pro iOS nemají takovouto orientaci, rozhodlo se, že aplikace KOSeek pro iOS musí mít tuto funkčnost. Takové rozhodnutí je uděláno kvůli tomu, že se taková orientace obrazovky určitě hodí pro zobrazení rozvrhu uživatele a jeho studijních výsledků.

1.8 Doménový model

Domain model (viz obrázek 1.16) popisuje základní třídy (entity) aplikace a vztahy mezi nimi.

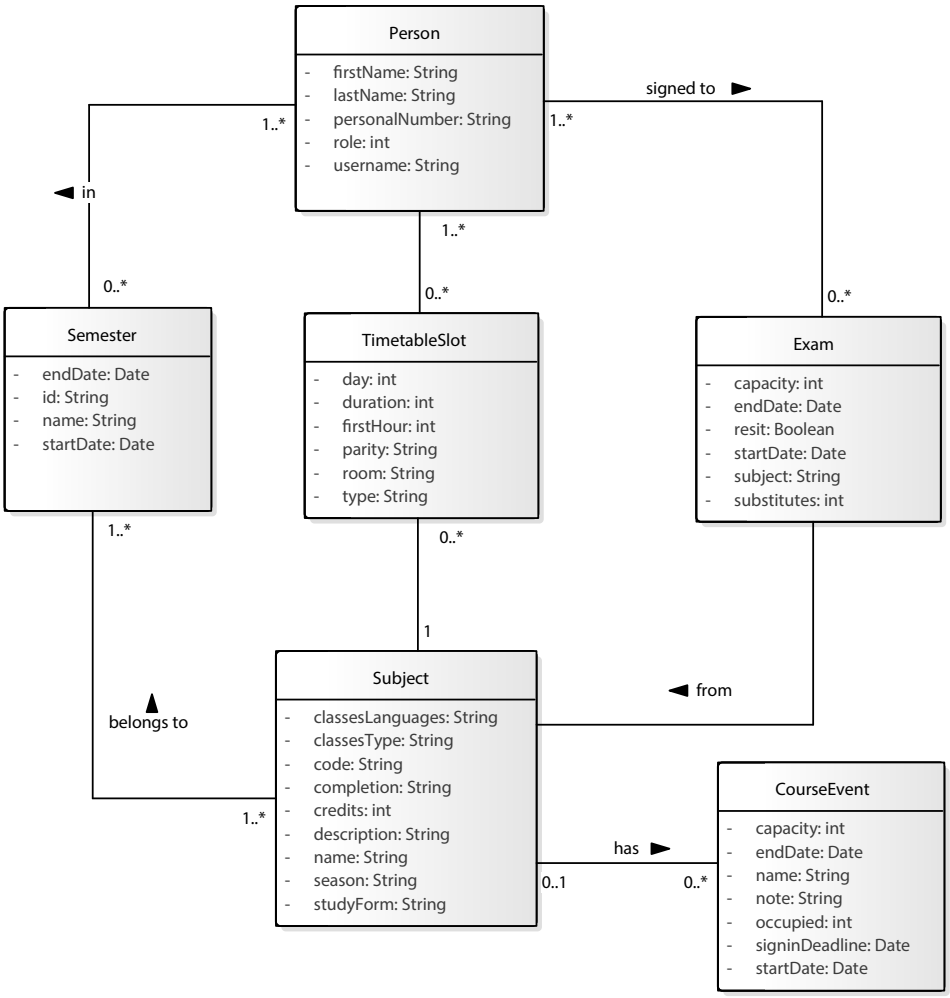
1.9 Procesy v aplikaci

Sekce popisuje některé procesy, které budou v aplikaci KOSeek pro iOS (application processes).

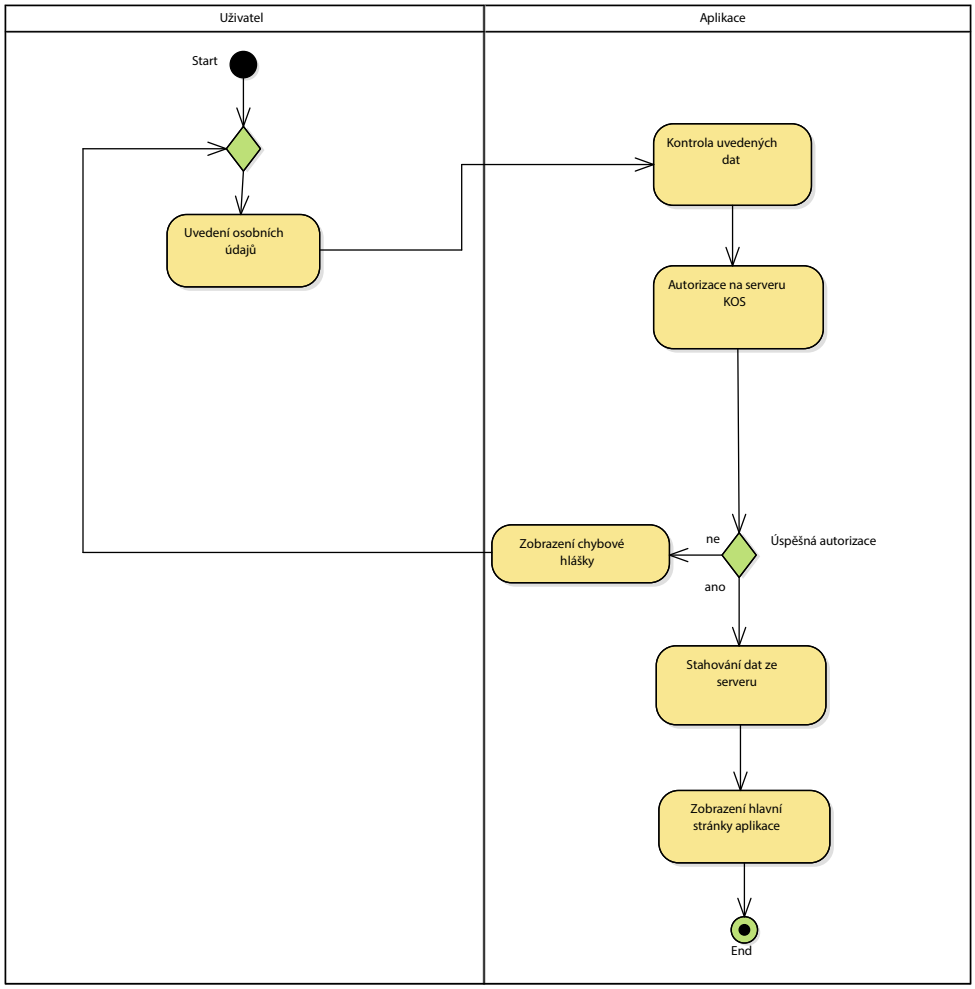
- Proces přihlášení (viz obrázek 1.17) — uživatel je při přihlašování do aplikace vyzván, aby zadal své uživatelské jméno a heslo. Po zkontrolovaní zadaných údajů proběhne pokus o autorizaci na straně serveru KOS. Pokud se uživatele nepodaří autorizovat, tak aplikace vypíše chybovou hlášku a znovu vyzve uživatele k zadaní uživatelského jména a hesla. V případě, že autorizace proběhne úspěšně, tak se přejde na hlavní stránku aplikace.
- Proces zobrazení zkoušek (viz obrázek 1.18) — uživateli je po zvolení položky zkoušky zobrazen seznam předmětů, které má zapsané v aktuálním semestru. Po zvolení konkrétního předmětu se aplikace pokusí získat a zobrazit online data. V případě, že se jí to nepodaří, tak je zobrazena chybová hláška.

1.10 Případy užití

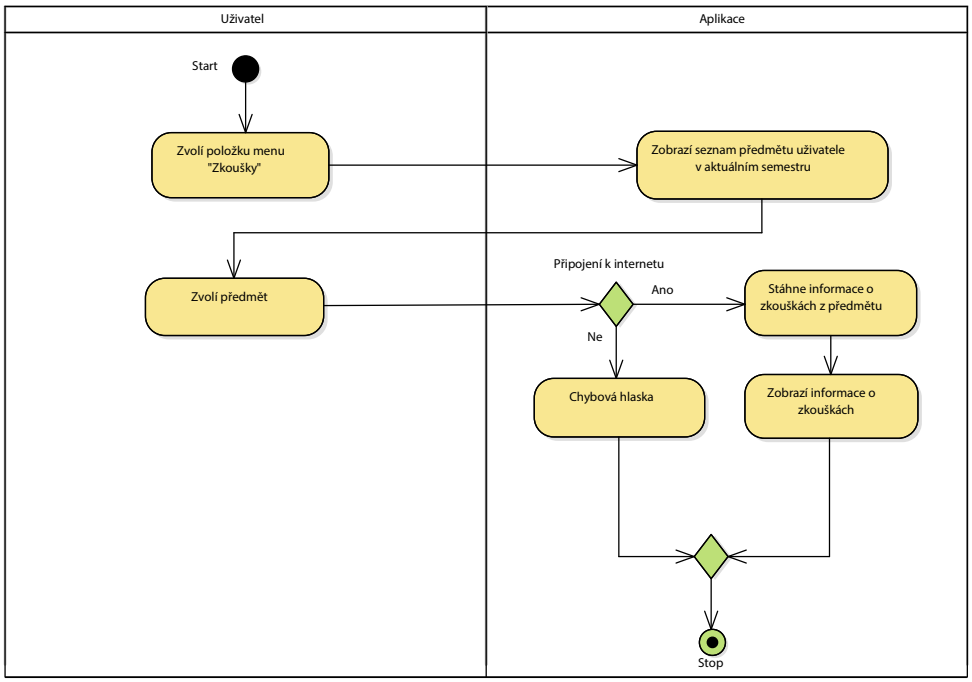
Případy užití aplikace jsou vytvořeny pouze pro rozvrh, zkoušky a předměty (viz obrázek 1.19). Ostatní případy užití (jednorázové akce, studijní oddělení) nejsou vytvořeny z důvodu opakování předchozích s jediným rozdílem v názvu. Účastník je pouze jeden — student. Účastník vyučující není zohledněn z důvodu nedostatečných přístupových práv k systému.



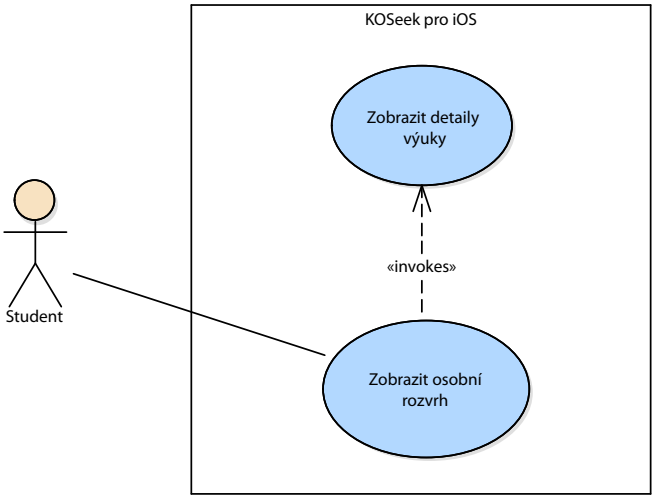
Obrázek 1.16: Doménový model



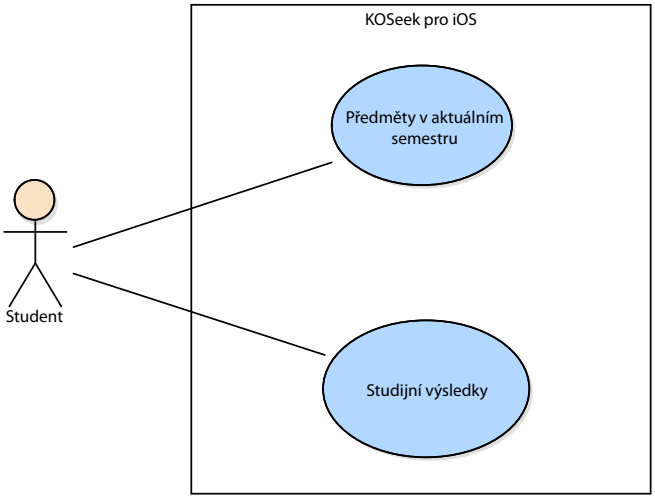
Obrázek 1.17: Proces přihlášení v aplikaci



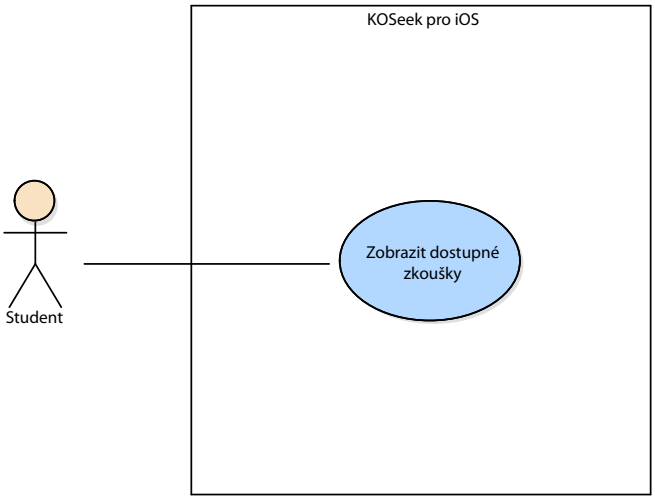
Obrázek 1.18: Proces zobrazení zkoušek v aplikaci



(a) Rozvrh



(b) Předměty



(c) Zkoušky

Obrázek 1.19: Případy užití

Kapitola 2

Návrh

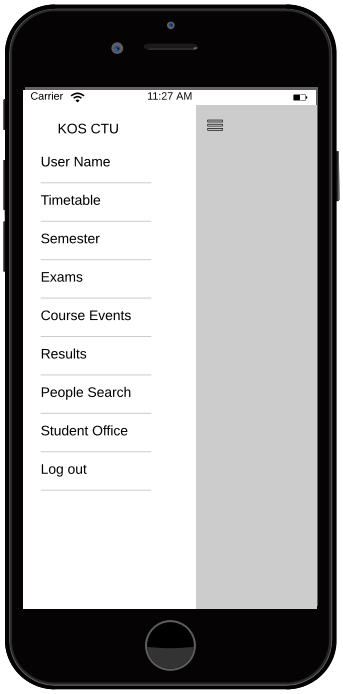
Kapitola popisuje návrh aplikace, který zahrnuje wireframy aplikace. Na základě analýz provedených v kapitole 1 je proveden návrh aplikace KOSeek pro iOS.

2.1 Wireframes

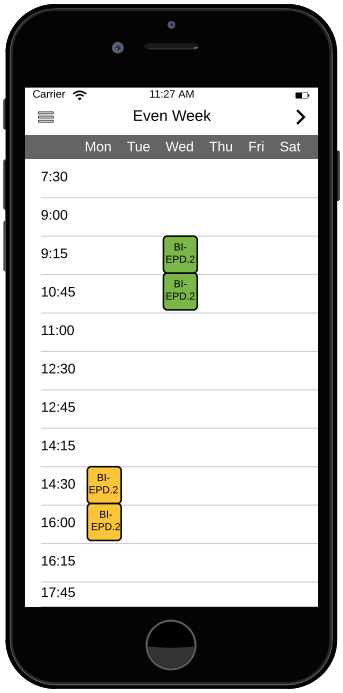
Wireframes je standardní technologie, která slouží k návrhu uživatelského rozhraní. Byl vytvořen návrh hlavního levého menu (viz obr. 2.1a), které lze otevřít gestem slide vpravo anebo příslušným tlačítkem. První funkci, kterou lze z menu vybrat je profil uživatele. Levé menu je tvořeno podle funkčních požadavků a kliknutím na název se otevře příslušná obrazovka.

Také jsou vytvořeny wireframy jednotlivých obrazovek aplikace (viz obr. 2.1). Šípkou na obrazovce rozvrhu (viz obr. 2.1b) uživatel může změnit paritu zobrazeného rozvrhu. Při stisknutí na předmět v rozvrhu se zobrazí alert, který obsahuje podrobné informace o výuce jako plný název, místnost, začátek a konec výuky, vyučující. Podle wireframu 2.1c ve studijních výsledcích je možné vybrat semestr dle názvu a zobrazí se všechny zapsané předměty v tomto semestru. Nahoře vlevo jsou informace o zapsaných a získaných kreditech ve vybraném semestru. Úspěšně zakončené předměty jsou označovány barvou. Pravá část této hlavičky obsahuje ty samé informace, ale to se týká celkového počtu kreditů ze studia.

Wireframe pro aktuální semestr není vytvořen kvůli tomu, že má skoro stejný smysl jako studijní výsledky s tím rozdílem, že se mají zobrazovat pouze informace o aktuálním semestru a chybí informace o kreditech v semestru. Ze stejných důvodů není udělán i wireframe pro jednorázové akce, který se má chovat podobně zkouškám. Rozdíl je v tom, že u jednorázových akcí uživatel nemá možnost vybrat kód předmětu podle kterého se má ukázat akce, tedy se zobrazí všechny aktuální jednorázové akce.

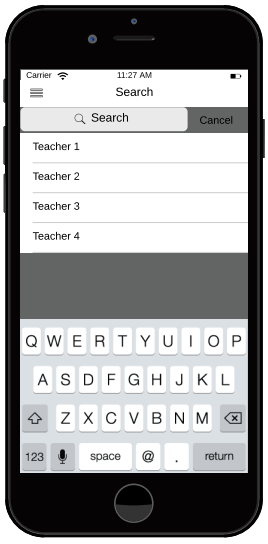


(a) Levé menu

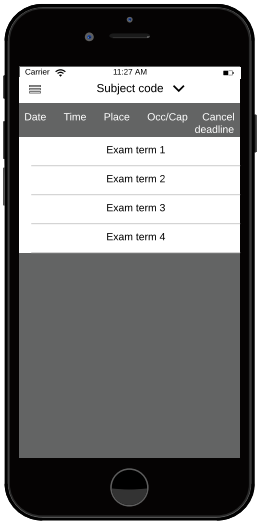


(b) Rozvrh

(c) Studijní výsledky



(d) Vyhledávání učitelů



(e) Zkoušky

Obrázek 2.1: Wireframes

2.2 Výběr technologií

2.2.1 Programovací jazyk

Zadání bakalářské práce nijak nespecifikuje programovací jazyk a proto je k dispozici několik variant, jelikož je aplikace mobilní a pro operační systém iOS. Při vývoji pro tento operační systém je možné programovat v těchto jazycích: Swift, Objective-C a C#. Rozhodlo se pro Swift kvůli níže uvedeným důvodům a také tomu, že v rámci předmětu BI-IOS se učí v tomto jazyce. Vývojové prostředí XCode je vybráno vzhledem k nejlepší podpoře a integraci jazyka Swift v tomto prostředí. Pro vhodnou práci nad aplikaci je zaveden systém pro správu verzí Git, který je bez problémů integrován do XCode IDE.

"*Swift is a powerful and intuitive programming language for iOS, OS X, and watchOS. Writing Swift code is interactive and fun, the syntax is concise yet expressive, and apps run lightning-fast*"[14]. Jazyk Swift je novým programovacím jazykem od společnosti Apple a je velkým přínosem pro vývojáře, kteří dělají software pro OS X a iOS. Hlavním plusem jazyka je jeho bezpečnost. Apple vytvořil tento jazyk s cílem, aby aplikace nespadla za dobu běhu a skoro všechny chyby byly objeveny za dobu kompilace.

2.2.2 Úložiště dat

Stažená data je potřeba ukládat a mít k nim rychlý přístup. Pro tyto účely je zvolen framework Core Data od Apple, který je popsán v sekci 1.6.2. „*Core Data is a framework that you use to manage the model layer objects in your application. It provides generalized and automated solutions to common tasks associated with object life cycle and object graph management, including persistence*“ [15]. Pomocí Core Data lze vytvořit model, kterým je možné popsat entity, jejich vlastnosti a relace mezi nimi. IDE XCode umožňuje rychlý způsob vytvoření takového modelu. Avšak Core Data není databází, to je pouze úložiště dat na zařízení, je tzv. objektovým grafickým správci.

Kapitola 3

Implementace a testování

Kapitola popisuje realizaci a testování aplikace KOSeek pro iOS a také jsou zde popsány použité knihovny a RESTful zdroje KOSapi.

3.1 Implementace

Aplikace je rozdělena na tři vrstvy: prezentační, aplikační a datová. Datová vrstva je reprezentována statickou třídou Database a třídami pro ukládání jednotlivých zdrojů z KOSapi. V aplikační vrstvě je nejdůležitější třídou LoginHelper, která provádí komunikaci se serverem pro přihlášení uživatele a získání access tokenu, případně jeho obnovení. Stahování všech dat v aplikaci umožňuje statická třída aplikační vrstvy KOSAPI pracující přímo s KOSapi. Stažená data tato třída ukládá prostřednictvím třídy Database. Za prezentaci dat jsou zodpovědné jednotlivé obrazovky jako osobní rozvrh, zapsané předměty v aktuálním semestru, studijní výsledky a vyhledávání učitelů. V iOS se používá tzv. model-view-controller návrhový vzor (viz sekce 1.6). Prezentační vrstvu aplikace tvoří view controllery, které komunikují s datovou vrstvou pro získání uložených dat a pro stahování dat pracují s aplikační vrstvou.

3.1.1 Knihovny

Aplikace je psaná v programovacím jazyce Swift ve vývojovém prostředí XCode. Knihovna SWXMLHash slouží k formátovaní a získání dat z XML souboru. Pro zarovnání GUI prvků jsou využity možnosti knihovny SnapKit. O levé menu a příslušnou animaci se stará Objective-C knihovna SWRevealViewController. Vypadávací menu je tvořeno pomocí knihovny BTNavigationDropDownMenu. Také pro práci s formátem JSON je využita knihovna Swifty-JSON. Všechny uvedené knihovny jsou přidány do projektu pomocí technologie pod souborů (podfile).

3.1.1.1 SWXMLHash

Aplikace KOSeek požaduje data z KOSu. Pomocí KOSapi lze získat data ze studijního systému KOS ve formátu XML. Pro rychlé zpracování stažených XML dat je použita knihovna SWXMLHash. Místo této knihovny je možné použit NSXMLParser třídu z iPhone SDK a také knihovnu AEXML. Rozhodlo se pro knihovnu SWXMLHash kvůli její snadné implementaci a použití. Ukázka parsování dat v aplikaci pomocí této knihovny:

```
private class func subjectDetailsParser(xml: XMLIndexer, context:
↪   NSManagedObjectContext) {
        let content = xml["atom:feed"]["atom:entry"]["atom:content"]
        let completion = content["completion"].element?.text
        let season = content["season"].element?.text
        let range = content["range"].element?.text
        let description = content["description"].element?.text
        let lecturesContents = content["lecturesContents"].element?.text
        let tutorialsContents =
↪   content["tutorialsContents"].element?.text
        Database.addSubjectDetails(code: subjectCode, completion:
↪   completion, range: range, season: season, description: description,
↪   lecturesContents: lecturesContents, tutorialsContents:
↪   tutorialsContents, context: context)
}
```

Listing 1: Příklad parsování detailů předmětu pomocí knihovny SWXMLHash

3.1.1.2 SWRevealViewController

Tato knihovna je napsána v jazyce Objective-C a hodí se pro vytváření levého menu. SWRevealViewController je view controller, který se stará o animace, rozpoznání gestů a navigaci. Nastavení jednotlivých obrazovek pro vracení do levého menu je potřeba udělat v kódu. V KOSeek pro iOS se to nastavuje v hlavní třídě v metodě viewDidLoad MainTableViewController (viz kód 2). Jednotlivé obrazovky pomocí dědění od hlavní třídy mají tuto vlastnost. Pro práci s knihovnou je nutné vytvořit root navigation controller a nastavit přechody mezi tímto view controllerem a obrazovkami, což je uděláno pomocí storyboard souborů.

3.1.2 RESTful zdroje

V této sekci jsou uvedeny všechny použité v aplikaci REST zdroje a typy obsahu.

```
override func viewDidLoad() {
    super.viewDidLoad()
    tableView.backgroundColor = TableViewBackgroundColor
    menuButton = UIBarButtonItem(image: UIImage(named: "menu"), style:
 ↪  UIBarButtonItemStyle.Plain, target: nil, action: nil)
    menuButton.tintColor = MenuButtonTintColor
    if self.revealViewController() != nil {
        menuButton.target = self
        menuButton.action = "menuButtonPressed"
        self.view.addGestureRecognizer(self.revealViewController().
 ↪  panGestureRecognizer())
    }
    self.navigationItem.leftBarButtonItem = menuButton
}

func menuButtonPressed() {
    childFuncBeforeShowSideMenu()
    UIApplication.sharedApplication().sendAction("revealToggle:",
     to: self.revealViewController(), from: self, forEvent: nil)
}
```

Listing 2: Vrácení do levého menu

3.1.2.1 CourseEvents

GET /courseEvents [16]

- URI: https://kosapi.fit.cvut.cz/api/3/courseEvents/
- Typ zdroje: Atom Feed
- Typ obsahu: CourseEvent
  - kapacita
  - předmět
  - název
  - počet obsazených míst
  - místo konání
  - semestr
  - začátek
- Formáty: atom, xml
- Parametry: fields, lang, limit, locEnums, multilang, offset, orderBy, query

3.1.2.2 Courses

GET /courses [17]

- URI: https://kosapi.fit.cvut.cz/api/3/courses/
- Typ zdroje: Atom Feed
- Typ obsahu: Course
  - kód
  - kredity
  - anotace
  - název
  - osnova přednášek
- Formáty: atom, xml
- Parametry: detail, sem, fields, lang, limit, locEnums, multilang, offset, orderBy, query

3.1.2.3 Divisions

GET /divisions [18]

- URI: https://kosapi.fit.cvut.cz/api/3/divisions/
- Typ zdroje: Atom Feed
- Typ obsahu: Division
  - kód
  - název
- Formáty: atom, xml
- Parametry: fields, lang, limit, locEnums, multilang, offset, orderBy, query

3.1.2.4 Exams

GET /exams [19]

- URI: https://kosapi.fit.cvut.cz/api/3/exams/
- Typ zdroje: Atom Feed
- Typ obsahu: Exam
  - uzávěrka odhlášení

  - kapacita
  - předmět
  - název
  - počet obsazených míst
  - místo konání
  - semestr
  - začátek
- Formáty: atom, xml
- Parametry: fields, lang, limit, locEnums, multilang, offset, orderBy, query

3.1.2.5 Semesters

GET /semesters/current [20]

- URI: https://kosapi.fit.cvut.cz/api/3/semesters/current
- Typ zdroje: Atom Entry
- Typ obsahu: Semester
  - datum ukončení
  - název
  - datum zahájení
- Formáty: atom, xml, text
- Parametry: fields, lang, limit, locEnums, multilang

3.1.2.6 Students

GET /students/studyCodeOrId/enrolledCourses [21]

- URI: https://kosapi.fit.cvut.cz/api/3/students/username/enrolledCourses/
- Typ zdroje: Atom Feed
- Typ obsahu: CourseEnrollment
- Formáty: atom, xml, text
- Parametry: sem, fields, lang, limit, locEnums, multilang, offset, orderBy, query

GET /students/studyCodeOrId/enrolledCourses [21]

- URI: https://kosapi.fit.cvut.cz/api/3/students/username
- Typ zdroje: Atom Feed
- Typ obsahu: Student
  - e-mail
  - křestní jméno
  - ročník studia
  - příjmení
  - osobní číslo
  - uživatelské jméno
- Formáty: atom, xml
- Parametry: fields, lang, locEnums, multilang

GET /students/studyCodeOrId/parallels [21]

- URI: https://kosapi.fit.cvut.cz/api/3/students/username/parallels/
- Typ zdroje: Atom Feed
- Typ obsahu: Parallel
  - kapacita
  - kód (číslo)
  - předmět
  - semestr
  - vyučující
  - uživatelské jméno

  TimetableSlot:
  - den
  - parita opakování
  - místnost
- Formáty: atom, xml
- Parametry: sem, fields, lang, limit, locEnums, multilang, offset, orderBy, query

3.1.2.7 Teachers

GET /teachers [22]

- URI: https://kosapi.fit.cvut.cz/api/3/teachers/
- Typ zdroje: Atom Feed
- Typ obsahu: Teacher
  - e-mail
  - křestní jméno
  - příjmení
  - uživatelské jméno
- Formáty: atom, xml
- Parametry: fields, lang, limit, locEnums, multilang, offset, orderBy, query

3.1.3 RESTful příklad

URI dotazu: `https://kosapi.fit.cvut.cz/api/3/students/username/enrolledCourses?access_token=accessToken&sem=semesterID&limit=1000&lang=language`

Použité proměnné:

- Username — uživatelské jméno.
- Access token získaný při autentizaci a autorizaci uživatele. Pokud platnost tokenu vypršela, aplikace získá nový access token pomocí refresh tokenu.
- Semester ID — id semestru podle kterého se mají stáhnout předměty.
- Language — jazyk stažených dat.

V aplikaci zdroj je použit pro získání studijních výsledků studenta. Odpověď ve formátu ATOM:

```
<?xml version="1.0" encoding="UTF-8"?>
<atom:feed xmlns="http://kosapi.feld.cvut.cz/schema/3"
 ↪  xmlns:atom="http://www.w3.org/2005/Atom"
 ↪  xmlns:osearch="http://a9.com/-/spec/opensearch/1.1/"
 ↪  xmlns:xlink="http://www.w3.org/1999/xlink"
 ↪  xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
 ↪  xml:base="https://kosapi.fit.cvut.cz/api/3" xml:lang="cs">
    <atom:id>https://kosapi.fit.cvut.cz/api/students/turlyalz/
 ↪  enrolledCourses</atom:id>
    <atom:updated>2016-03-25T11:50:16.781</atom:updated>
```

```
    <atom:link rel="next"
↪    href="students/turlyalz/enrolledCourses?access_token=
↪    1abd43bb-8cbb-44d6-a9ea-859beef2f657
↪    &amp;sem=none&amp;lang=cs&amp;offset=5&amp;limit=2"/>
    <atom:entry>
        <atom:id>urn:cvut:kos:courseenrollment:415132202305</atom:id>
        <atom:updated>2013-01-23T02:09:38.0</atom:updated>
        <atom:author>
            <atom:name>hyniova</atom:name>
        </atom:author>
        <atom:content atom:type="xml"
↪    xsi:type="internalCourseEnrollment">
            <assessment>true</assessment>
            <completed>true</completed>
            <extern>false</extern>
            <semester xlink:href="semesters/B121/">Zimní
↪    2012/2013</semester>
            <course xlink:href="courses/BI-CAO/">Číslicové a analogové
↪    obvody</course>
        </atom:content>
    </atom:entry>
    <atom:entry>
        <atom:id>urn:cvut:kos:courseenrollment:415132202505</atom:id>
        <atom:updated>2013-02-06T18:42:57.0</atom:updated>
        <atom:author>
            <atom:name>trlifkat</atom:name>
        </atom:author>
        <atom:content atom:type="xml"
↪    xsi:type="internalCourseEnrollment">
            <assessment>true</assessment>
            <completed>true</completed>
            <extern>false</extern>
            <semester xlink:href="semesters/B121/">Zimní
↪    2012/2013</semester>
            <course xlink:href="courses/BI-MLO/">Matematická
↪    logika</course>
        </atom:content>
    </atom:entry>
    <osearch:startIndex>0</osearch:startIndex>
    <osearch:itemsPerPage>2</osearch:itemsPerPage>
</atom:feed>
```

3.1.4 LoginHelper třída

Statická třída aplikace poskytující metody pro získání a obnovení access tokenu, pomocí kterého probíhá komunikace se serverem a získání zdrojů KOSapi. Přetížená metoda getAuthToken se volá okamžitě po stisknutí tlačítka login s parametry username a password. Tato funkce postupně volá statické metody pro kontrolu správnosti uživatelských údajů a jeho oprávnění (viz 3 metoda loginRequest), provádění autentizace, získání kódu a access tokenu. Vlákno

v této metodě čeká ve smyčce dokud probíhá komunikace anebo dokud není překročena maximální doba čekání, která je stanovena na 12 sekund. Pokud dojde k chybě během komunikaci, například nesprávné údaje uživatele, tak se vrátí chybová hláška, která je zobrazena uživateli v příslušném view controlleru. Druhá varianta metody getAuthToken, která se volá bez parametrů vrátí access token, pokud je platný, jinak obnovuje token pomocí metody refreshAuthToken. Metody se většinou volají ve vlákně, které běží na pozadí, aby při komunikaci nedocházelo k blokování například čekacího GUI elementu. Jednotlivé endpointy, které jsou použity v aplikaci jsou popsány v 1.2.4. sekci.

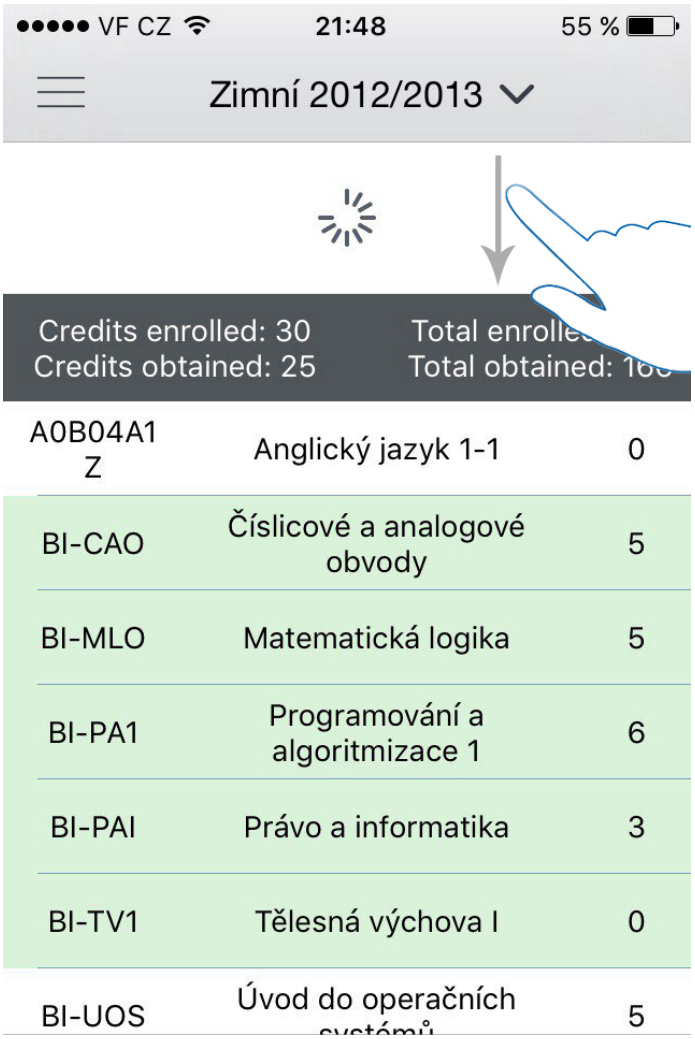
```
private class func loginRequest(username: String, password: String) ->
↪   (success: Bool, error: String) {
    let request = NSMutableURLRequest(URL: NSURL(string: baseURL +
↪   login)!)
    request.HTTPMethod = "POST"
    let postString = "j_username=" + username + "&j_password=" +
↪   password
    var failed = false
    var running = false
    request.HTTPBody =
↪   postString.dataUsingEncoding(NSUTF8StringEncoding)
    let task = NSURLSession.sharedSession().dataTaskWithRequest(request)
↪   {
        data, response, error in
        if error != nil {
            failed = true
            return
        }
        running = false
    }
    running = true
    task.resume()

    var count = 0
    while running && !failed && count < MaxWaitForResponse {
        sleep(1)
        count++
    }

    if failed || errorOcurredIn(task.response) || count >=
↪   MaxWaitForResponse {
        return (false, "Log in failed. Please try again.")
    }

    return (true, "200 OK")
}
```

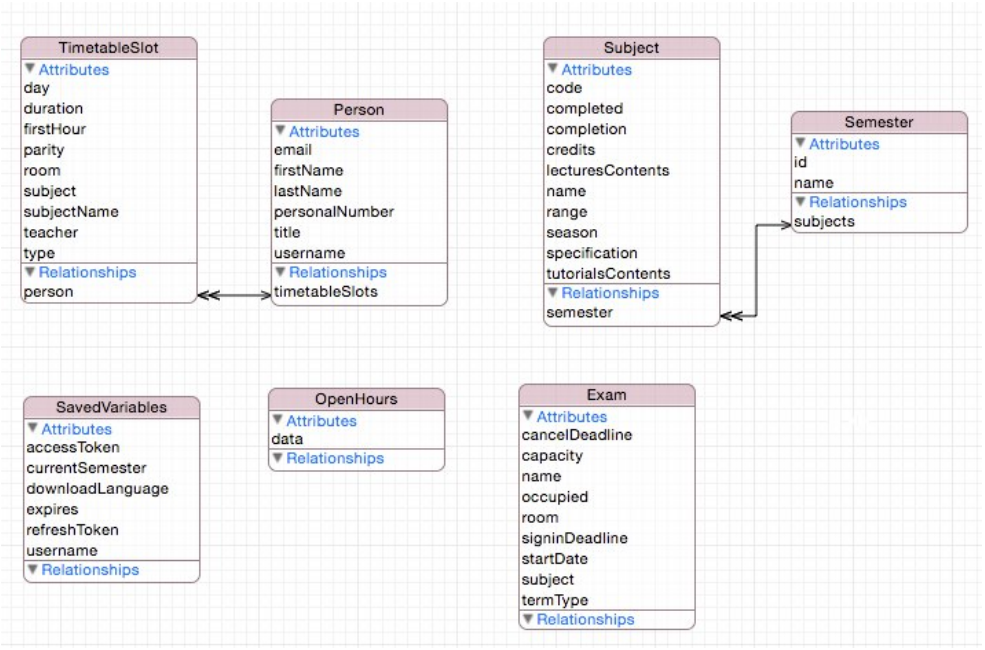
Listing 3: Metoda kontrolující uživatelské údaje a jeho oprávnění



Obrázek 3.1: Funkce pull to refresh

3.1.5 KOSAPI třída

Tato statická třída obsahuje metody pro stahování potřebných zdrojů prostřednictvím KOSapi. Metoda downloadAllData stahuje všechny zdroje a je volána po úspěšné autorizaci a autentizaci uživatele. Také metoda se volá když uživatel stiskne tlačítko obnovení všech dat. V podstatě downloadAllData volá všechny metody pro stahování jednotlivých zdrojů. Kromě toho KOSAPI třída má metody pro parsování získaných RESTful zdrojů pomocí knihovny SWXMLHash. Ukázka parsování je uvedená zde 1. Jednotlivé zdroje se aktualizují když uživatel táhne dolů obrazovkou (pull to refresh, viz obrázek 3.1).



Obrázek 3.2: Model entit Core Data

3.1.6 Uložiště dat

Stažená data se ukládají pomocí frameworku od Apple — Core Data. Na obrázku 3.2 je Core Data model KOSeek aplikace. Relace entit jsou popsány pouze tam, kde to je potřeba. Pokud je nutné přidat další relaci, lze toto jednoduše provést. Entita SavedVariables je použita pro účely ukládání těchto dat: access token, aktuální semestr, refresh token, uživatelské jméno a jiné.

V aplikaci ukládání a získání dat pomocí Core Data umožňuje třída Database. Tato statická třída poskytuje funkce pro smazání, uložení, získání dat podle různých kritérií. Funkce getSemesterBy hledá semestr podle parametru id a vrací instanci třídy Semester (viz kód 4). Používá se první element z result kvůli jednoznačnosti id semestru.

Pro správné pracování s Core Data rozhraním v hlavním vlákně a vlákně, které běží na pozadí, jsou vytvořeny dvě pomocné třídy: CoreDataHelper a CoreDataStore. CoreDataHelper poskytuje dva různé kontexty (pro hlavní vlákno a vlákno běžící na pozadí). Také třída poskytuje metodu pro ukládání dat na základě předaného kontextu. Druhá třída poskytuje tři vlastnosti: odkaz na adresář, kde se nachází Core Data Store soubor, Managed Object Model a Persistent Store Coordinator (používá se pro přístup k databázovému souboru).

```
class func getSemesterBy(id: String, context: NSManagedObjectContext) ->
↪  Semester? {
    let request = NSFetchRequest(entityName: "Semester")
    request.returnsObjectsAsFaults = false
    request.predicate = NSPredicate(format: "id == %@", id)
    do {
        let results = try context.executeFetchRequest(request)
        if results.count > 0 {
            return results[0] as? Semester
        }
    }
    catch let error as NSError {
        debugPrint(error)
    }
    return nil
}
```

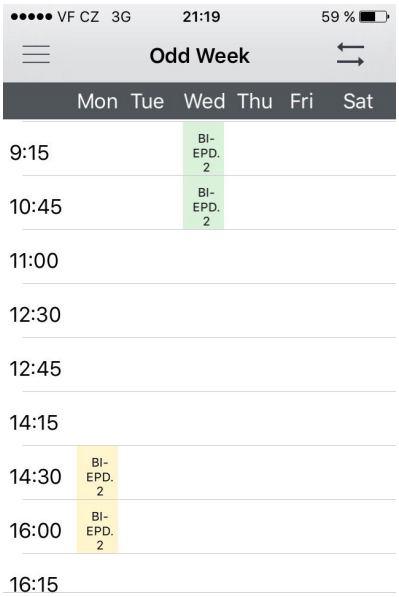
Listing 4: Metoda pro vracení semestru dle jeho id

3.1.7 Prezentační vrstva aplikace

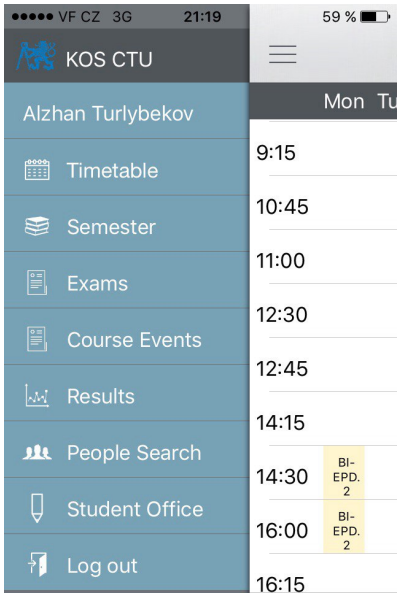
Prezentační vrstvou v aplikaci jsou jednotlivé view controllery, které pomocí třídy Database získávají data a následně je zobrazují. Všechny tyto view controllery mají společného předka standardní třídu UITableViewController, která poskytuje view controllerům zobrazení dat v buňkách. Jednotlivé obrazovky lze vidět na obrazcích 3.3. Profil uživatele obsahuje nastavení jazyka stahování (viz obr. 3.3d). Důvodem k vytvoření takovéhoto tlačítka je ten fakt, že není jasné jaký si uživatel preferuje jazyk stažených dat. Vzhledem k tomu, že KOSapi poskytuje pouze anglickou a českou verze dat, stejně jako KOS, tak například někdo s ruským jazykem systému může chtít stahovat data v českém a někdo naopak v anglickém jazyce. Při kliknutí na jednotlivou výuku v rozvrhu se zobrazuje alert obsahující podrobné informace o výuce (viz obr. 3.4a).

3.1.8 Lokalizace Aplikace

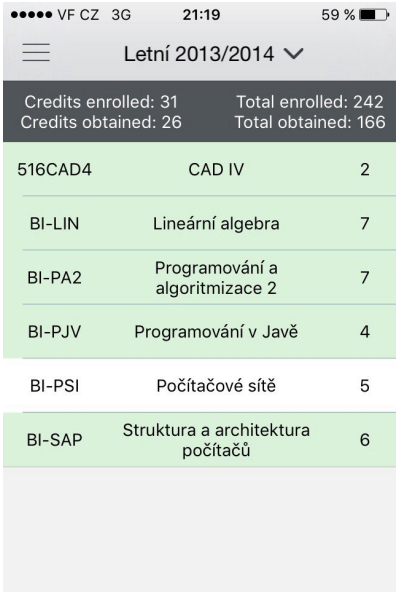
XCode IDE poskytuje vhodnou technologii pro lokalizaci aplikace, což lze provést exportováním XML souboru a následnou jeho úpravou pro každý jazyk. XCode automaticky vyexportuje všechna slova použité ve storyboard souborech. Avšak slova použita v kódu je potřeba vytvořit pomocí třídy NSLocalizedString. Všechna takováto slova jsou vytvořena v souboru LocalizableStrings. Podle nefunkčních požadavků jsou vytvořeny překlady do českého a ruského jazyků.



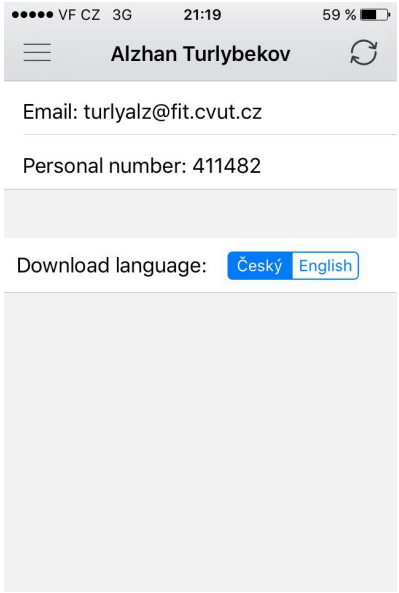
(a) Rozvrh



(b) Levé menu

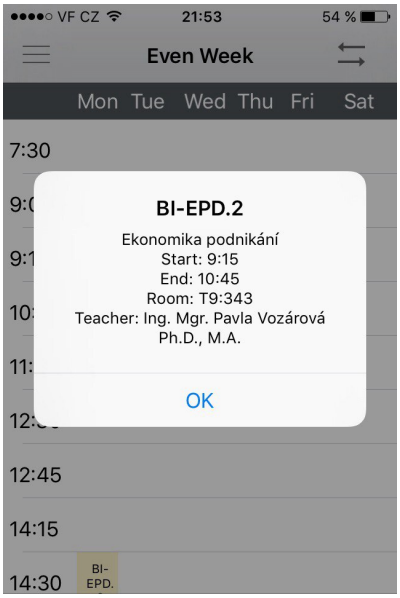


(c) Studijní výsledky

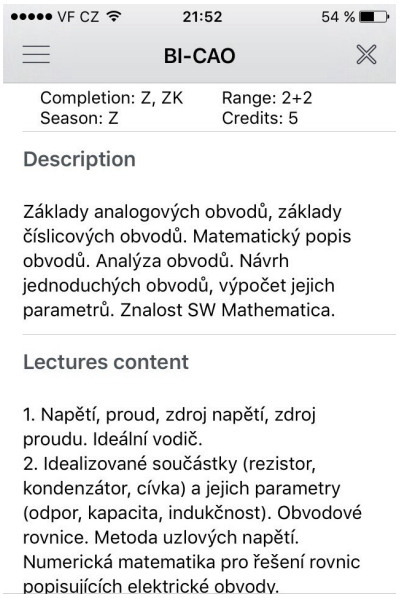


(d) Profil uživatele

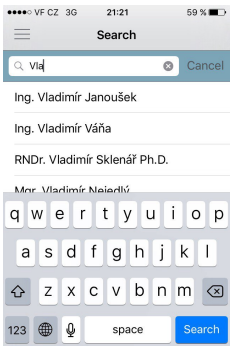
Obrázek 3.3: Obrazovky výsledné aplikace



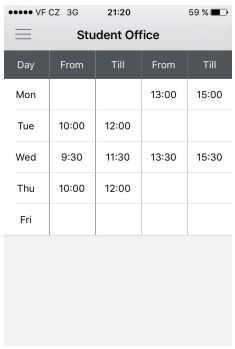
(a) Detaily paralelky v rozvrhu



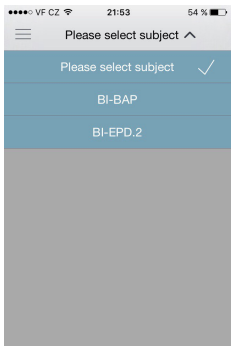
(b) Detaily předmětu



(c) Vyhledávání učitelů



(d) Studijní oddělení



(e) Výběr předmětu

Obrázek 3.4: Obrazovky výsledné aplikace

3.2 Testování

3.2.1 Testování na zařízeních

Během vývoje aplikace byla neustále testována na iPhone 4S s operačním systémem iOS 9.0. Dále aplikace byla testována na emulátorech iPhone 6, iPhone 6S, IPad 2.

3.2.2 Testování na vybrané skupině uživatelů

Vybraná skupina ze čtyř lidí zjistila následující problémy a vylepšení aplikace:

- Zapomenutý kód předmětů u jednorázových akcí.

- Chybějící emaily vyhledaných učitelů.
- Přidání tlačítka pro zavření detailů předmětů a vracení do obrazovky studijní výsledky anebo aktuální semestr.

Všechny nalezené chyby jsou úspěšně opraveny a všechna vylepšení jsou zapracována do aplikace.

3.2.3 Testování UI

XCode poskytuje tzv. XCUI API, které umožňuje interakci s aplikací tak, jakoby to dělal uživatel. Také XCode podporuje psaní testů pomocí zachycení interakce s aplikací. UI testování řeší dva úkoly: testuje zda aplikace funguje a také zda-li je přístupná [23]. Bylo napsáno několik funkcí uvnitř třídy KOSeekUITests pro testování UI podle případů užití. Příklad UI testu, který ověřuje správnost vyhledávání učitelů:

```
func testSearch() {
    let app = XCUIApplication()
    let tablesQuery = app.tables
    tablesQuery.staticTexts["People Search"].tap()
    let typeToStartSearchField = tablesQuery.searchFields["type to
↪   start"]
    typeToStartSearchField.tap()
    typeToStartSearchField.typeText("vag")
    XCTAssertEqual(app.tables.cells.elementBoundByIndex(0).staticTexts.
↪   elementBoundByIndex(0).label, "Ing. Ladislav Vagner Ph.D.")
    tablesQuery.buttons["Cancel"].tap()
    typeToStartSearchField.tap()
    typeToStartSearchField.typeText("Nonvalid name")
    XCTAssertTrue(app.tables.cells.count == 0)
}
```

3.2.4 Funkční testy

Tyto testy mají za cíle ověřit realizaci funkčních požadavků. Většinou testy obsahují následující požadavky:

- Suitability (vhodnost)
- Accuracy (přesnost)
- Interoperability (interoperabilita)
- Compliance (vyhovění)
- Security (bezpečnost)

Toto testování bylo provedeno v rámci unit testů a během psaní kódu.

3.2.5 Unit testy

Jedná se o testování jednotlivých tříd a metod. Unit testy pomohly najít několik závažných chyb v aplikaci. Bylo využito testovací prostředí, které nabízí XCode pomocí třídy XCTestCase, děděním od které lze využit její funkcionality. Metoda setUp se volá před spuštěním každého testu a metoda tearDown po jeho spuštění. Také je k dispozici test výkonu, který měří čas provádění bloku kódu. Jednotlivé metody pro testování se musí začínat slovem test, aby XCode poznal, že se jedná o test. Bylo napsáno několik testovacích tříd, které testují odpovídající třídy v aplikaci. Ukázka třídy KOSAPITests, která testuje parsování a stahování dat v aplikaci:

```
func testDownload() {
    XCTAssert(Database.getSavedVariablesFrom(context:
↪   SavedVariables.cdh.managedObjectContext).currentSemester != nil, "")
}

func testKOSAPIdata() {
    XCTAssert(LoginHelper.getAuthToken() != nil, "Access token is nil!")
    KOSAPI.download("Timetable slots", extensionURL: "/students/" +
↪   SavedVariables.username! + "/parallels?access_token=" +
↪   LoginHelper.getAuthToken()! + "&limit=1000&lang=cs", context:
↪   SavedVariables.cdh.managedObjectContext, parser: parser)
}

func parser(xml: XMLIndexer, context: NSManagedObjectContext) {
    if let slotNumber = KOSAPI.getNumberFrom(xml) {
        for index in 0...slotNumber-1 {
            let content =
↪   xml["atom:feed"]["atom:entry"][index]["atom:content"]
            let subject =
↪   content["course"].element?.attributes["xlink:href"]?.
↪   stringByReplacingOccurrencesOfString("courses/",
↪   withString:"").stringByReplacingOccurrencesOfString("/", withString:
↪   "")
            XCTAssert(subject != nil, "No subject!")
            let subjectName = content["course"].element?.text
            XCTAssert(subjectName != nil, "No subject name!")
            let type = content["parallelType"].element?.text
            XCTAssert(type != nil, "No type!")
            let teacher = content["teacher"].element?.text
            XCTAssert(teacher != nil, "No teacher!")
            let slot = content["timetableSlot"]
            let dayStr = slot["day"].element?.text
            XCTAssert(dayStr != nil, "No day!")
        }
    } else {
        XCTAssert(true, "No timetable slots")
    }
}
```

Kapitola 4

Vyhodnocení

Aplikace KOSeek pro iOS je přínosem pro studenty, kteří chtějí využívat aplikaci KOS přes mobilní zařízení s operačním systémem iOS. Výsledná aplikace má rychlou odezvu a podle funkčních požadavků lze aplikaci využívat v režimu offline. Podle mého názoru je výsledná aplikace velmi shodná s její návrhem. Všechny funkční a nefunkční požadavky na aplikaci byly splněny a také byly splněny všechny stanovené na začátku cíle.

4.1 Možnosti rozšíření

KOSapi a usermapApi poskytují hodně informací, o které je možné aplikaci rozšířit. V první řadě to může být vyhledávání předmětů, případně seznam předmětů, které si může student zapsat na příští semestr. Také vhodným rozšířením aplikace mohou být rozvrhy učitelů a více informací o nich. Jednorázové akce v KOSu obsahují další informace, které mohou být užitečné. Tyto informace je možné zobrazovat při kliknutí na jednotlivou akci. Co se týká kódu, tak vylepšením je refaktorizace některých tříd, například TimetableViewController je možné podědit od třídy UICollectionViewController a tím už nebude nutné ručně kreslit jednotlivé paralelky studenta.

Závěr

V rámci této bakalářské práce byla vytvořena aplikace KOSeek pro iOS, která slouží k vyhledávání kontaktů ve škole, informacím o studijním oddělení a předmětech. Také práce popisuje operační systém iOS, jeho strukturu a možnosti. Dále byly popsány možnosti KOSapi, pomocí kterého probíhá komunikace a získávání dat z KOSu.

Práce obsahuje návrh a analýzu aplikace včetně funkčních a nefunkčních požadavků a wireframy jednotlivých obrazovek. KOSeek pro iOS umožňuje nahlížet do konkretních dat studijního informačního systému KOS, jako například zapsané předměty studenta, jeho studijní výsledky a rozvrh. Kromě toho pomocí aplikace lze zobrazit aktuální zkoušky a jednorázové akce v semestru dle kódu předmětu. Uživatel má možnost kdykoliv obnovit buď všechna data aplikace anebo jednotlivá data obrazovky pomocí funkce pull to refresh. Pak následuje jednotlivý popis implementace a testování aplikace.

Doufám, že aplikace bude užitečná a vhodná pro studenty s mobilním operačním systémem iOS. Lze aplikaci rozšiřovat o další funkcionalitu, což bylo popsáno v poslední kapitole práce. Přínosem aplikace je to, že danou aplikaci můžu předat týmu studentů ČVUT, který bude rozšiřovat a starat se o aplikaci a nebude muset začít od začátku. Tímto doufám, že práce bude přínosem nejenom pro můj vlastní rozvoj a rozšíření mých vědomostí, ale také pro další studenty ČVUT.

Literatura

[1] ČVUT: Komponenta studium (KOS) [online]. [cit. 2016-02-15]. Dostupné z: http://intranet.cvut.cz/informace-pro-studenty/is/kos

[2] Jirůtka, J.: Projekt KOSapi v3.2 [online]. [cit. 2016-02-13]. Dostupné z: https://kosapi.fit.cvut.cz/projects/kosapi/wiki

[3] Fredrich, T.: *RESTful Service Best Practices*. Pearson eCollege, 2012, [cit. 2016-02-14]. Dostupné z: http://www.restapitutorial.com/media/RESTful_Best_Practices-v1_1.pdf

[4] Jirůtka, J.: KOSapi — RESTful zdroje [online]. [cit. 2016-02-13]. Dostupné z: https://kosapi.fit.cvut.cz/projects/kosapi/wiki/Resources

[5] Jirůtka, J.: OAuth 2.0. 2014, [cit. 2016-02-26]. Dostupné z: https://rozvoj.fit.cvut.cz/Main/oauth2

[6] Jenčík, D.: KOSeek [online]. © 2015, [cit. 2016-03-21]. Dostupné z: http://dusanjencik.cz/portfolio/koseek/

[7] Univerzita Karlova v Praze: Studijní informační systém — Návod k používání SIS [online]. © 2016, [cit. 2016-03-22]. Dostupné z: https://is.cuni.cz/studium/help.php?modul=stev

[8] Apple Inc.: About the iOS Technologies [online]. © 2014, [cit. 2016-02-29]. Dostupné z: https://developer.apple.com/library/ios/documentation/Miscellaneous/Conceptual/iPhoneOSTechOverview/Introduction/Introduction.html

[9] Apple Inc.: Core OS Layer [online]. © 2014, [cit. 2016-03-01]. Dostupné z: https://developer.apple.com/library/ios/documentation/Miscellaneous/Conceptual/iPhoneOSTechOverview/CoreOSLayer/CoreOSLayer.html//apple_ref/doc/uid/TP40007898-CH11-SW1

[10] Apple Inc.: Core Services Layer [online]. © 2014, [cit. 2016-03-03]. Dostupné z: https://developer.apple.com/library/ios/documentation/Miscellaneous/Conceptual/iPhoneOSTechOverview/CoreServicesLayer/CoreServicesLayer.html//apple_ref/doc/uid/TP40007898-CH10-SW5

[11] Apple Inc.: Media Layer [online]. © 2014, [cit. 2016-03-03]. Dostupné z: https://developer.apple.com/library/ios/documentation/Miscellaneous/Conceptual/iPhoneOSTechOverview/MediaLayer/MediaLayer.html//apple_ref/doc/uid/TP40007898-CH9-SW4

[12] Apple Inc.: Cocoa Touch Layer [online]. © 2014, [cit. 2016-02-29]. Dostupné z: https://developer.apple.com/library/ios/documentation/Miscellaneous/Conceptual/iPhoneOSTechOverview/iPhoneOSTechnologies/iPhoneOSTechnologies.html

[13] Apple Inc.: The App Life Cycle [online]. © 2015, [cit. 2016-03-07]. Dostupné z: https://developer.apple.com/library/ios/documentation/iPhone/Conceptual/iPhoneOSProgrammingGuide/TheAppLifeCycle/TheAppLifeCycle.html

[14] Apple Inc.: Swift [online]. © 2016, [cit. 2016-03-03]. Dostupné z: https://developer.apple.com/swift/

[15] Apple Inc.: What Is Core Data? [online]. © 2016, [cit. 2016-02-18]. Dostupné z: https://developer.apple.com/library/tvos/documentation/Cocoa/Conceptual/CoreData/index.html

[16] Jirůtka, J.: KOSapi — Jednorázové akce předmětů [online]. [cit. 2016-02-22]. Dostupné z: https://kosapi.fit.cvut.cz/projects/kosapi/wiki/CourseEvents

[17] Jirůtka, J.: KOSapi — Předměty [online]. [cit. 2016-02-22]. Dostupné z: https://kosapi.fit.cvut.cz/projects/kosapi/wiki/Courses

[18] Jirůtka, J.: KOSapi — Střediska [online]. [cit. 2016-02-22]. Dostupné z: https://kosapi.fit.cvut.cz/projects/kosapi/wiki/Divisions

[19] Jirůtka, J.: KOSapi — Zkouškové termíny [online]. [cit. 2016-02-22]. Dostupné z: https://kosapi.fit.cvut.cz/projects/kosapi/wiki/Exams

[20] Jirůtka, J.: KOSapi — Semestry [online]. [cit. 2016-02-22]. Dostupné z: https://kosapi.fit.cvut.cz/projects/kosapi/wiki/Semesters

[21] Jirůtka, J.: KOSapi — Studenti [online]. [cit. 2016-02-22]. Dostupné z: https://kosapi.fit.cvut.cz/projects/kosapi/wiki/Students

[22] Jirůtka, J.: KOSapi — Vyučující [online]. [cit. 2016-02-22]. Dostupné z: https://kosapi.fit.cvut.cz/projects/kosapi/wiki/Teachers

[23] Sherman, J.: UI Testing in Xcode 7 [online]. © 2016, [cit. 2016-03-26]. Dostupné z: https://www.bignerdranch.com/blog/ui-testing-in-xcode-7-part-1-ui-testing-gotchas/

Příloha A

Seznam použitých zkratek

API Application Programming Interface

ATOM Atom Syndication Format

BSD Berkeley Software Distribution

FEL Fakulta elektrotechnická

FIT Fakulta informačních technologií

GUI Graphical user interface

HTTP Hypertext Transfer Protocol

IDE Integrated Development Environment

iOS Iphone operating system

JSON JavaScript Object Notation

MIDI Musical Instrument Digital Interface

REST Representational State Transfer

RSS Really Simple Syndication

SDK Software Development Kit

URI Uniform Resource Identifier

VPN Virtual Private Network

XML Extensible markup language

ČVUT České vysoké učení technické

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