Sem vložte zadání Vaší práce.

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

Bakalářská práce

Časový vyhledávač lidí a místností na FIT

Jan Staněk

Vedoucí práce: Ing. Ondřej Guth, Ph.D

11. května 2015

Poděkování

Tímto bych chtěl poděkovat svému vedoucímu Ing. Ondřeji Guthovi, Ph.D za konzultace a rady při realizaci této práce. Dále bych rád poděkoval přítelkyni a rodině za nekončící podporu.

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 11. května 2015 .....

Odkaz na tuto práci

Staněk, Jan. *Časový vyhledávač lidí a místností na FIT*. Bakalářská práce. Praha: České vysoké učení technické v Praze, Fakulta informačních technologií, 2015.

Abstrakt

Tato bakalářská práce se zabývá vývojem mobilní aplikace pro Android, jejímž primárním cílem je zobrazování rozvrhových událostí studentů a učitelů na Fakultě informačních technologií ČVUT a volných místností. Aplikace čerpá data z aplikačních rozhraní Sirius a KOSapi. V aplikaci bylo použito OAuth 2.0 protokolu za účelem autorizace.

Práce postupně popisuje jednotlivé fáze vývoje aplikace od analýzy, návrhu přes realizaci až po testování. Nakonec je popsán směr, kterým bude aplikace v budoucnu směřovat. Výsledek práce je funkční mobilní aplikace s prototypem uživatelského rozhraní.

Klíčová slova Vyhledávání na FIT, aplikace na Android, uživatelské rozhraní, rozvrhové události, Projekt Sirius, KOSapi, OAuth2.0 autorizace

Abstract

This bachelor thesis describes development of a mobile application for Android. Its primary goal is to show schedule events which belong to students and teachers from Faculty of Information Technology CTU and empty rooms. The data source of the application comes from application interfaces Sirius and KOSapi. For the purpose of authorization the OAuth 2.0 protocol was used.

The thesis describes each stage of an application development. These stages are analysis, design, implementation and a testing stage. In the end there is mentioned a path which will be taken by the application in the future. The result of the thesis is a functional mobile application with a prototype of an user interface.

Keywords Searching at FIT, Android application, user interface, schedule events, Project Sirius, KOSapi, OAuth2.0 authorization

Obsah

Seznam obrázků

Úvod

Mnoho přístupných místností se nachází na Fakultě informačních technologií Českého vysokého učení technického v Praze, kde některé zejí prázdnotou, zatímco se studenti mačkají v respiriu. Někteří nesnášejíce zadýchané prostory se vydávají na okružní cestu po místnostech, které fakulta využívá. Avšak někdy se stává, že z důvodu svátků se například v úterý učí podle čtvrtečního rozvrhu a naopak. To už student pro jistotu, aby se nezúčastnil hodiny navíc, raději prchá se učit do knihovny, kde ho žádné podobné překvapení potkat nemůže.

A co teprve, když se student rozhodne najít učitele, aby s ním mohl osobně probrat důležité informace k semestrální práci. To bez napsání mailu či po delším hledání na internetu nejde.

Tématem mé bakalářské práce je aplikace, která v sobě skloubí funkce vyhledávání lidí a volných místností na fakultě informačních technologií. Aplikace bude s uživateli komunikovat pomocí jednoduchého a přehledného uživatelského rozhraní, což hledání zpříjemní a zefektivní.

Motiv práce

Od té doby, co jsem zjistil, že určité učebny jsou v době, kdy se v nich neučí, přístupné pro studenty, tak jsem se již respiriím vyhýbal obloukem. Je to mnohem pohodlnější, klidnější a člověk tam nemusí trpět dusno. Avšak tyto volné místnosti je nutné najít. Hledání volné učebny člověku zabere v průměru několik minut. Buď se postupně obejde každá místnost, kterou Fakulta informačních technologií na ČVUT spravuje a jednoduše se najde ta, kde nikdo není, nebo se využije internetu. Nicméně ne málokrát se stala věc, že místnost podle internetu obsazená zela prázdnotou a naopak.

Hledání informace, kde mi probíhá hodina, která mi má začít za 4 minuty. To by se dalo říct, byl můj denní chleba. K této informaci jsem většinou přistupoval přes stránku Timetables. Avšak ta vám místo této informace zobrazí váš 14 denní rozvrh a vám nezbývá nic jiného, než si místo konání vyhledat v něm. Tudíž v obou případech jsem pravidelně ztrácel cenné minuty.

Další věcí, která mě motivovala k vytvoření nového systému, bylo přispění něčím vlastním do komunity ČVUT, což by při kladném přijetí zpříjemnilo

život jak studentům, tak i učitelům.

Cíl práce

Cílem práce je navrhnout a vhodným způsobem implementovat aplikaci, která vyřeší zmíněné problémy. Zmíněnými problémy jsou: zobrazení volných místností, aktuálních a platných událostí, které uživatele čekají a vyhledávání rozvrhů učitelů. Informace budou přístupné i v momentě, kdy uživatel nebude připojen k internetu. Aplikace bude využívat již vytvořených prostředků v rámci infrastruktury FIT ČVUT. Velká priorita bude věnována uživatelskému rozhraní tak, aby bylo přívětivé a intuitivní.

Kapitola 1

Analýza

V této kapitole bych chtěl popsat současná řešení, která se potýkají se zmíněnou problematikou a jsou volně dostupná. Poté problémy zachytím ve funkčních a nefunkčních požadavcích a kapitolu uzavřu případy užití.

1.1 Současný stav řešení problematiky

Při hledání současných řešení problematiky jsem zejména procházel již zaběhnutými systémy na ČVUT a k tomu jsem hledal systémy nově vzniklé. Pátral jsem po aplikacích, které nějakým způsobem pracují s rozvrhem studenta na FIT, spravují místnosti na FIT či jen poskytují informace ohledně zkoušek studentů. Hledal jsem především přes vyhledávač Google a na stránkách s mobilními aplikacemi a to na Google Play a Windows Phone Store. Níže jsou popsané systémy, které jsem buď znal, nebo našel hledáním. U každého popíši, co je jeho hlavním úkolem a jak se u něj dá nebo nedá dostat k chtěným informacím. Nakonec všechny problémy, a jestli je systémy řeší, shrnu v tabulce.

1.1.1 Timetable

Timetables je internetová stránka, která primárně slouží ke zjištění rozvrhu pro kohokoliv z FIT a pro jakoukoliv místnost, ve které se vyučují předměty pod záštitou fakulty. Na stránku se dostane kdokoliv, kdo se připojuje ze sítě FIT, avšak pro získání informací odjinud je vyžadováno přihlášení přes SSO bránu. Stránka čerpá svá data denně z informačního systému KOS [1].

Timetables mě po přihlášení vpustí na hlavní stránku, která slouží jako rozcestník. Cesty, kterými se můžu vydat, jsou rozvrhy předmětů, místností, studentů, učitelů a seznam zkoušek. Pro získání svého rozvrhu musím po přejití na stránku se seznamem studentů najít své jméno a to rozkliknout. K nalezení nejbližší hodiny, která mě čeká, musím znát aktuální čas a údaj, zdali je právě probíhající týden sudý nebo lichý, jelikož tato informace zde chybí. Pro vyhledání volné místnosti je třeba vybrat aktuální den opět s ohledem na li-

chost týdnů. Poté musím manuálně projít seznamem místností s jejich rozvrhy a hledat díry v rozvrhu podle zrovna probíhající vyučovací hodiny. Timetables sice zobrazují termíny zkoušek, avšak nezobrazují mnou přihlášenou zkoušku.

Výhodou tohoto systému je to, že se zde dají najít všechny rozvrhy, kterými fakulta oplývá a pro již zkušeného uživatele za počítačem je snadné najít požadovanou informaci do minuty. Nevýhodou naopak je, že pokud se uživatel bude chtít okamžitě dozvědět, kde má hodinu, na kterou jde už teď pozdě, musí buď vytáhnout počítač nebo se zdlouhavě neprakticky proklikat na chtěnou informaci za pomoci chytrého telefonu. Při této situaci je nalezení svého jména mezi ostatními studenty na malé obrazovce náročný úkol a překliky na sebe nenechají dlouho čekat. Tudíž při hledaní rozvrhové události, kdy na uživatele tlačí čas, se Timetable neosvědčí. Nakonec člověku ještě neznalému této stránky může pod přehršlí funkcí trvat delší dobu, než se na stránce zorientuje.

1.1.2 Studijní informační systém KOS

Ve studijním informačním systému KOS se především vytváří vlastní rozvrh, přihlašuje ke zkouškám a jednorázovým akcím a kontroluje studijní plán[2]. Na rozdíl od výše zmíněného Timetable, KOS zobrazuje informace jenom o přihlášeném. Přihlášení do systému probíhá přes SSO bránu nebo přes ČVUT login. Musím podotknout, že KOS je tu s námi již přes dvě desítky let a webová aplikace podle toho i vypadá[3]. Navíc po narážkách ze strany učitelů je mezi akademickou obcí ČVUT značně známo, že KOS není zejména v rámci databáze navržený tak, jak by měl, a tudíž je u něj obdivuhodné, že vydržel do dneška.

Co se týče průchodu systémem, tak ten začíná, když mě systém po přihlášení přivítá archaicky vypadající stránkou, na jejíž homepage se vyskytuje můj krátký medailonek a seznam nadcházejících fakultních událostí. Z hlavní stránky se můžu dostat buď k osobnímu rozvrhu, ke studijním výsledkům, či se můžu přihlásit ke zkoušce anebo si zapsat předmět. Osobní rozvrh a studijní výsledky jsou lehce dostupné a k přihlášeným zkouškám se doklikám také celkem rychle. Při zobrazení rozvrhu se dostanu pouze k informacím ohledně vyučujícího, takže vyčíst, kde se v tu dobu nachází někdo jiný z FIT, nedokážu.

Za výhodu můžeme považovat, že své dosavadní studijní výsledky nikde jinde nenajdeme tak uspořádané a u sebe jako tady. Nevýhodu vidím zejména v zastaralosti webové aplikace a pro uživatele získávajícího informace převážně z mobilu, je KOS zkouškou trpělivosti. Další nevýhodou je nepřizpůsobující se rozvrh nečekaným změnám, jako jsou např. záměny čtvrteční výuky za úterní z důvodu svátků.

1.1.3 ČVUT FIT – Kalendář

Dalším řešením je ČVUT kalendář, který vznikl z iniciativy Google Apps for Education[4]. Každý takto může sdílet svůj denní rozvrh mezi všemi lidmi z FIT, díky čemuž se každý z fakulty může podívat, kde se zrovna osoby sdílející svůj rozvrh nachází. Co se týče rozvrhů místností, tak ty Google Calendar nespravuje žádným způsobem.

Do svého kalendáře se dostanu po přihlášení svým ČVUT loginem. Hned po autentifikaci se dostanu na stránku s kalendářem, kde jsou označeny všechny události ostatních a události vlastní, které jsem si do kalendáře sám přidal. Pro přidání dalšího kalendáře jednoduše najdu hledanou osobu v ostatních kalendářích. I import vlastního rozvrhu je snadný a to díky Projektu Sirius, jelikož stačí pouze zadat adresu Siria se svým rozvrhem.

Výhodou kalendáře je to, že pokud se o něj každý stará, poskytuje aktuální informace. Z toho vyplývá i jeho nevýhoda a to je, že uživatel musí o kalendář pečovat a pravidelně ho poctivě plnit, což potencionální uživatele dokáže odradit. Další nevýhodou je, že kalendář nevyužívá každý, a tudíž ne každý by se dal najít podle tohoto řešení. Ovšem věc k dobru je ta, že uživatel se pro získání chtěné informace nemusí proklikávat složitou strukturou stránek. Stačí jenom zadat adresu s kalendářem, a pakliže si uživatel nastavil všechny rozvrhy, které chce sledovat, informaci získá ihned.

1.1.4 KOSapi

KOSapi poskytuje přístup k vybrané části databázi KOS přes RESTful aplikační rozhraní. KOSapi umožnuje vznik školních i studijních aplikací online zpřístupněním dat souvisejících s výukou, čímž si zajistil první místo jakožto zdroje dat rozvrhových událostí[5]. Pro získání informací je nutné se prvně přihlásit ČVUT loginem. Data se posléze dají získat ve formátu Atom, což je formát založený na XML, který se používá k syndikaci obsahu[6].

1.1.5 Projekt Sirius

Dalším systémem, který spravuje rozvrhové události je již zmíněný Projekt Sirius. Projekt vznikl v rámci ČVUT v roce 2013 a realizuje rozhraní pro přístup k rozvrhovým datům ČVUT. Data poskytuje buď ve formátu iCalendar nebo ve formátu JSON. Čerpá je z KOSapi a poté z nich vytváří rozvrhové události, které ještě upravuje podle různých rozvrhových pravidel a výjimek, které má v sobě uložené[7][8]. Projekt Sirius je v podstatě ideální řešení pro získávání informací, které jsem si v úvodu vytyčil. Avšak Sirius je serverová aplikace, která poskytuje svá data jen přes své API, a tudíž nenabízí žádné uživatelské rozhraní, které by funkčnost zpřístupňovalo uživatelům.

Výhodou této aplikace je aktuálnost, která zaručuje, že kdykoliv, kdy budete čerpat data ze Siria, měli byste mít vždy přesné a platné informace. Na druhou stranu, jak jsem již zmínil, aplikace je serverová, a proto ho uži-

vatelé využívají jen pro import kalendářů do aplikace Google Calendar. Nedostatkem Siria je to, že neobsahuje informace o jednorázových akcích studentů, či kolonky info u zkoušek, do které učitelé s oblibou píší důležité poznámky. Je ale nutné podotknout, že Projekt Sirius je k aktuálnímu datu ještě nedokončený a že mnohé funkce se postupem času objeví.

1.1.6 KOSeek

KOSeek je mobilní aplikace na Android a Windows Phone, jejímž hlavním účelem je poskytovat manuál ČVUT a je zejména určena pro lidi působící na FEL a FIT[9]. Aplikace dokáže zobrazit kontakt na učitele a studenty, zobrazí i rozvrh a informace o předmětech. Dále poskytuje informace o úředních hodinách studijního oddělení a i to, co se dneska dává k obědu v menze, ovšem po kliknutí na ikonu nás aplikace pouze odkáže na stáhnutí jiné aplikace.

Aplikace bez nutnosti přihlášení nabízí hned na úvodní stránce možnost podívat se na obecné informace ke studiu na Fakultě informačních technologií a na Fakultě elektrotechnické na ČVUT i na jejich novinky. Po kliknutí na kalendář zobrazí důležitá data, jako je například datum odevzdání bakalářské práce či rektorský den. Aplikace poskytuje vyhledávání osob na zmíněných fakultách a zobrazení jejich rozvrhu. V rozvrhu se vyskytuje i informace jestli zrovna probíhá týden sudý, či lichý. Dále aplikace usnadňuje vyhledávání automatickým napovídáním jména hledané osoby. Po přihlášení se svým ČVUT loginem se k hlavním stránkám přidá i stránka s vlastním profilem, kde najdu informace o sobě, mnou zapsané hodiny i dosavadní výsledky. Navíc je zde možnost zapisovat si poznámky a přidat si často hledané osoby do oblíbených. Aplikace podporuje odhlášení od účtu.

Výhodou této aplikace je bezesporu množství informací, které zde můžeme najít ohledně studia obecně. Osobní rozvrh má pěkné uživatelské rozhraní, nicméně pokud by uživatel měl mnoho předmětů v prvních dnech týdne, tak než by se dostal k informacím, kde se zrovna teď, v pátek ve 3 hodiny nachází, trvalo by to delší dobu. Další výhodou je i to, že aplikace ukládá stažená data do databáze, a tudíž není třeba stálého internetového připojení. Aplikace bohužel nespravuje nijak volné místnosti ani přihlášené termíny zkoušek.

1.1.7 Kosaak

Kosaak je v podstatě velmi odlehčená verze KOSeeku, která měla sloužit jako jeho odnož na mobilní operační systém Windows Phone. Z KOSeeku vzala jenom část s osobním profilem, kde se dají zobrazit informace o sobě, o zapsaných předmětech a o vlastním rozvrhu a všechno ostatní dala bokem[10]. Na první pohled aplikace působí čistěji a přehledněji s moderně vypadajícím uživatelským rozhraním.

Výhodou oproti KOSeeku je tedy větší přehlednost. Další výhodou je to, že při stisknutí tlačítka rozvrhu se automaticky zobrazí aktuální den v týdnu,

což dosti urychlí hledání. K ničemu dalšímu se bohužel využít nedá.

1.1.8 ČVUT navigátor

Primárním cílem této mobilní aplikace je ulehčení orientace nových studentů v prostorách školy a mezi různými okolními služby poskytujícími budovami. Systém kromě jiného nabízí i osobní rozvrh[11]. I když je aplikace zatím v beta vývoji, je možné si ji vyzkoušet přes obchod Windows Store. Bohužel se na ní k dnešnímu dni nejde přihlásit, a tudíž máme jen omezený přístup k veškerým funkcím.

Aplikace po spuštění vybídne uživatele k přihlášení. To se mi bohužel nepovedlo a co jsem se dočetl v diskuzi na Windows Store, nebyl jsem sám. Nicméně jsem i přes to měl možnost prozkoumat, jak vypadá to, co mě zajímá z aplikace nejvíce, osobní rozvrh. Osobní rozvrh je řešen tak, že zobrazuje klasickou tabulku se dny uspořádanými pod sebou a vyučujícími hodinami uspořádanými vedle sebe. Tento typ rozvrhu považuji jako nevýhodu, neboť přes celou obrazovku nejde vidět více jak čtvrtina rozvrhu a uživatel musí chvilku brouzdat, aby se dopátral, jestli se něco děje ve čtvrtek odpoledne. Navíc, co jsem viděl z obrázků na profilové stránce aplikace na Windows Phone, rozvrh sice zobrazí, co uživatel v určitou hodinu má, avšak už neukáže, kde se to koná. Stále však je tam možnost, že se tato informace skrývá pod kliknutím na políčko s hodinou, to ovšem bez přihlášení nezjistíme.

1.1.9 Shrnutí

Postupně jsem zmínil snad většinu dostupných řešení problému s rychlostí vyhledání informace, volnými místnostmi, přihlášenými zkouškami a rozvrhy. Žádné z popsaných současných řešení nebylo pro zadané cíle dostatečnou odpovědí. Pro shrnutí splněných a nesplněných cílů si vezmeme ty systémy, které disponují uživatelským rozhraním, a tudíž jsou aktivně využívány jako cílové systémy. Tyto systémy posléze srovnáme v tabulce1.1. Tabulka pro každý systém ohodnotí, zda poskytuje uživateli pohled na vlastní rozvrh, dokáže vyhledat a zobrazit rozvrhy ostatních, a jestli ukáže přihlášené zkoušky či seznam volných místností. Dále jestli ihned zobrazí nejbližší hodinu a jestli poskytnou informace i bez připojení k internetu. V poslední řadě, jestli dokáže uživatel získat informace z těchto systému v rámci pár sekund ve výtahu.

1.2 Analýza požadavků

Jednou z nejdůležitějších věcí, bez které se neobejde žádná větší aplikace, jsou správně sepsané požadavky na systém. V knize [12] se zmiňují, že nedostatečně specifikované požadavky a nedostatečné zapojení uživatelů jsou dvěma hlavními příčinami konečného neúspěchu projektu[1]. Požadavky si rozdě-

[1] Arlow, J.; Neustadt, I.: UML 2 a unifikovaný proces vývoje aplikací, s. 78

	Timetable	KOS	Kalendář	KOSeek	Kosaak	Navigátor
Vlastní rozvrh	ano	ano	ano	ano	ano	ano
Rozvrhy ostatních	ano	ne	ano	ano	ne	ne
Přihlášené zkoušky	ne	ano	ne	ne	ne	ne
Volné místnosti	ano	ne	ne	ne	ne	ne
Aktuální událost	ne	ne	ano	ne	ano	ne
Přístup offline	ne	ne	ano	ano	ano	–
Rychlý přístup	ne	ne	ano	ano	ano	ne

Tabulka 1.1: Srovnání systémů s uživatelským rozhraním

líme na funkční a nefunkční, podle nichž se budeme snažit o co nepřesnější zachycení požadavků na systém. Nejprve však zmíním slovíčka, která budu v následujících odstavcích využívat.

1.2.1 Slovníček pojmů

- **Filtrování** Filtrování výsledků spočívá v selekci těch výsledků, které odpovídají kritériím.
- **Našeptávání** Našeptávání se objevuje ve chvíli, kdy uživatel musí sám přesně vyplnit textové pole. Uživateli se po zadání minimálně 3 znaků zobrazí seznam možností, které obsahují jím zadaný řetězec znaků. V této chvíli si uživatel buď vybere z nabídky, nebo bude pokračovat v psaní, čímž se bude zužovat i seznam nabízených možností. Nakonec si uživatel buď vybere možnost z nabídky, nebo text dopíše.

1.2.2 Funkční požadavky

První skupinou jsou požadavky funkční, které formulují požadované funkce systému.

F1 Vyhledání rozvrhových událostí studenta/učitele podle času
Systém bude poskytovat možnost vyhledání rozvrhových událostí studenta popř. učitele. Uživatel bude mít možnost specifikovat čas, odkdy a dokdy má události hledat. Při zadávání jména popř. uživatelského jména, bude systém uživateli našeptávat. Každá rozvrhová událost bude zobrazovat kód předmětu, čas konání a místnost, kde událost bude probíhat. Navíc bude zobrazovat typ události. Typy událostí jsou:

1. Zkouška
2. Přednáška
3. Cvičení
4. Laboratoř

Systém posléze vypíše seznam všech událostí seřazený vzestupně podle času událostí.

F2 Vyhledání volné učebny podle zadaných specifikací
Aplikace zobrazí seznam učeben, které budou odpovídat zadaným specifikacím a nebudou obsazené. Jestliže žádná specifikace nebude uvedena, aplikace zobrazí všechny aktuálně volné učebny. Typy specifikací jsou:

- Podle času
- Podle potřebné kapacity
- Podle typu učebny
    1. Počítačová učebna
    2. Seminární učebna
    3. Přednášková místnost

U každé volné místnosti bude zobrazena informace o typu učebny, kódu učebny a času, dokdy bude místnost neobsazená. Systém posléze vypíše seznam volných místností seřazený alfabeticky vzestupně podle kódu místností.

F3 Zobrazení přihlášených zkoušek
Systém bude poskytovat možnost zobrazit seznam přihlášených zkoušek přihlášeného uživatele. U každé zkoušky bude zobrazen datum a místo, kdy a kde se zkouška bude konat. Zkouška bude označená kódem předmětu, který zkouší. Seznam bude seřazený vzestupně podle data zkoušek.

F4 Zobrazení obsazenosti místností
Systém bude zobrazovat seznam místností, kde každá místnost bude dávat najevo, zda je obsazená či volná. Pokud bude místnost volná, zobrazí se čas dokdy bude volná, pakliže bude obsazená, zobrazí se čas, odkdy bude volná. Navíc bude místnost zobrazovat maximální kapacitu a typ místnosti. Seznam bude seřazen alfabeticky vzestupně podle kódu místnosti.

F5 Zobrazení rozvrhových událostí místnosti
Systém bude poskytovat možnost zobrazit rozvrhové události místnosti stejným způsobem jako je to u studentů a učitelů. Jediný rozdíl bude, že se zobrazí jenom ty události, které se budou konat daný den.

F6 Přidání popř. odebrání místnosti do popř. ze seznamu místností
Systém bude poskytovat možnost přidat nebo odebrat místnost ze seznamu místností. Při přidávání systém bude našeptávat místnosti, tak aby uživatel nemusel vyplňovat celý kód.

F7 Zobrazení nejbližší rozvrhové události přihlášeného uživatele
Systém bude zobrazovat přímo na hlavní obrazovce nejbližší rozvrhovou událost, která čeká přihlášeného uživatele.

F8 Zobrazení nejbližší přihlášené zkoušky přihlášeného uživatele
Systém bude zobrazovat přímo na hlavní obrazovce nejbližší zkoušku, která čeká přihlášeného uživatele.

F9 Odhlášení uživatele
Systém bude poskytovat možnost odhlásit uživatele a přihlásit se pod jiným ČVUT loginem. Pro každý login se bude ukládat seznam hledání v rychlé volbě.

F10 Filtrování volných místností
Systém bude podporovat filtrování volných místností podle typu učebny. Výsledek filtrování bude opět seřazen alfabeticky vzestupně podle kódu místností.

F11 Filtrování rozvrhových událostí Systém bude podporovat filtrování v seznamu rozvrhových událostí podle typu události. Výsledek filtrování bude seřazen vzestupně podle data.

F12 Vyhledávání podle rychlé volby
Systém bude poskytovat možnost vyhledání volné místnosti či osoby z FIT včetně specifikací za pomocí rychlé volby. Rychlá volba bude seznam několika posledních hledání, které učinil uživatel. Rychlá volba bude řadit hledání podle data uskutečnění hledání a podle toho, zda jsou připnuté. Pokud uživatel provede další hledání v okamžiku, kdy seznam rychlé volby je již plný, nahradí se první nepřipnuté hledání novým.

F13 Připnutí/Odepnutí hledání k/z seznamu rychlé volby
Systém bude podporovat připínání a odepínání provedeného hledání. Pokud hledání bude připnuté, žádné další hledání ho nenahradí.

F14 Odebrání hledání z rychlé volby
Systém bude poskytovat možnost odebrat hledání z rychlé volby.

F15 Manuální aktualizace seznamu učitelů a studentů na FIT
Systém bude poskytovat možnost manuálně aktualizovat seznam učitelů a studentů na FIT.

F16 Manuální aktualizace rozvrhových událostí
Systém bude poskytovat možnost manuálně aktualizovat seznam rozvrhových událostí.

1.2.3 Nefunkční požadavky

Druhou skupinou jsou požadavky nefunkční, které formulují podmínky omezující systém.

N1 Dvojjazyčné provedení
Systém bude proveden ve dvou jazykových verzích. Jazyky budou:

- Čeština
- Angličtina

N2 Získávání rozvrhových dat z aplikace Sirius
Systém bude získávat rozvrhová data z již zmíněné serverové aplikace Sirius.

N3 Rozvrhová data budou přístupná i offline
Aplikace bude zobrazovat již získaná data i bez připojení na internet.

N4 Rozšiřitelnost a modifikovatelnost
Aplikace bude implementována s důrazem na objektové modelování, tak aby byla v budoucnu snadno rozšiřitelná doplňky, které nebudou v rozporu s filosofií aplikace samotné.

N5 Intuitivní uživatelské rozhraní
Aplikace bude mít moderní, příjemné a intuitivní uživatelské rozhraní.

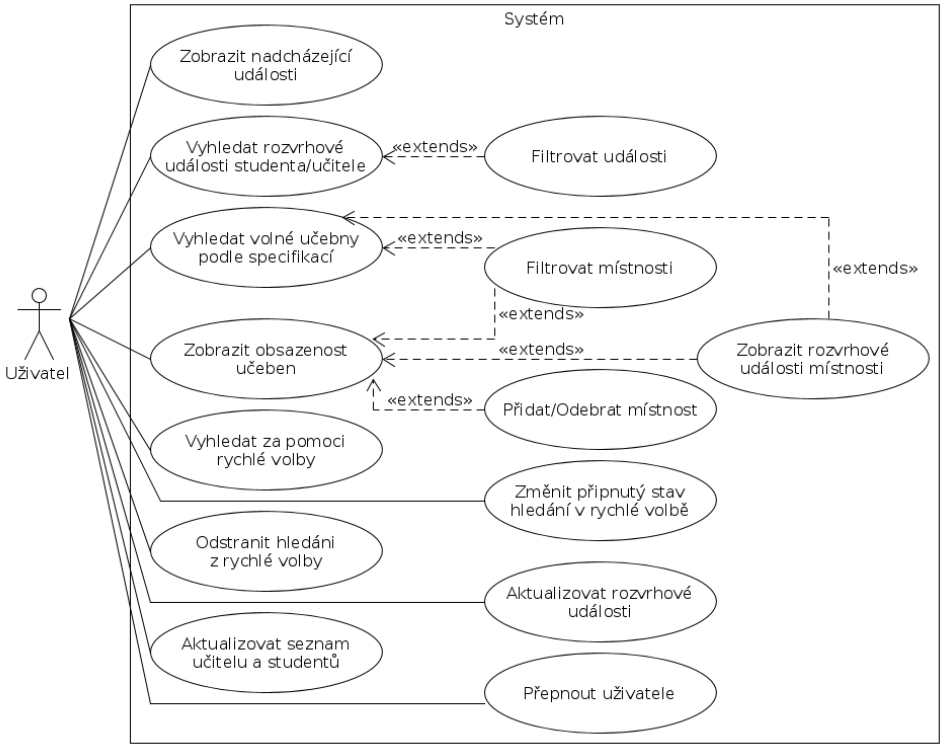
1.3 Případy užití

Pro jasnější přehled, co vlastně uživatel bude moci s aplikací udělat nám poslouží diagram případů užití, který je znázorněn na obrázku 1.1. Zajímavé případy rozepíši slovně.

UC1 Zobrazit nadcházející události
Umožňuje uživateli zobrazit nejbližší rozvrhové události, které ho čekají.
Scénář: Uživatel si zobrazí seznam událostí, které ho čekají.
Postup bude pro svou složitost vyobrazen v aktivity diagramu 1.2. Scénář začíná v okamžiku, kdy je uživatel přihlášen do aplikace, nachází se na úvodní obrazovce.

UC2 Vyhledat rozvrhové události studenta/učitele
Umožňuje vyhledání rozvrhových událostí studenta nebo učitele na FIT.

UC3 Vyhledání volné učebny podle specifikací
Umožňuje uživateli si najít volné místnosti, které budou splňovat jeho specifikace. Specifikace se týkají typu učebny, potřebné kapacity a minimální doby,

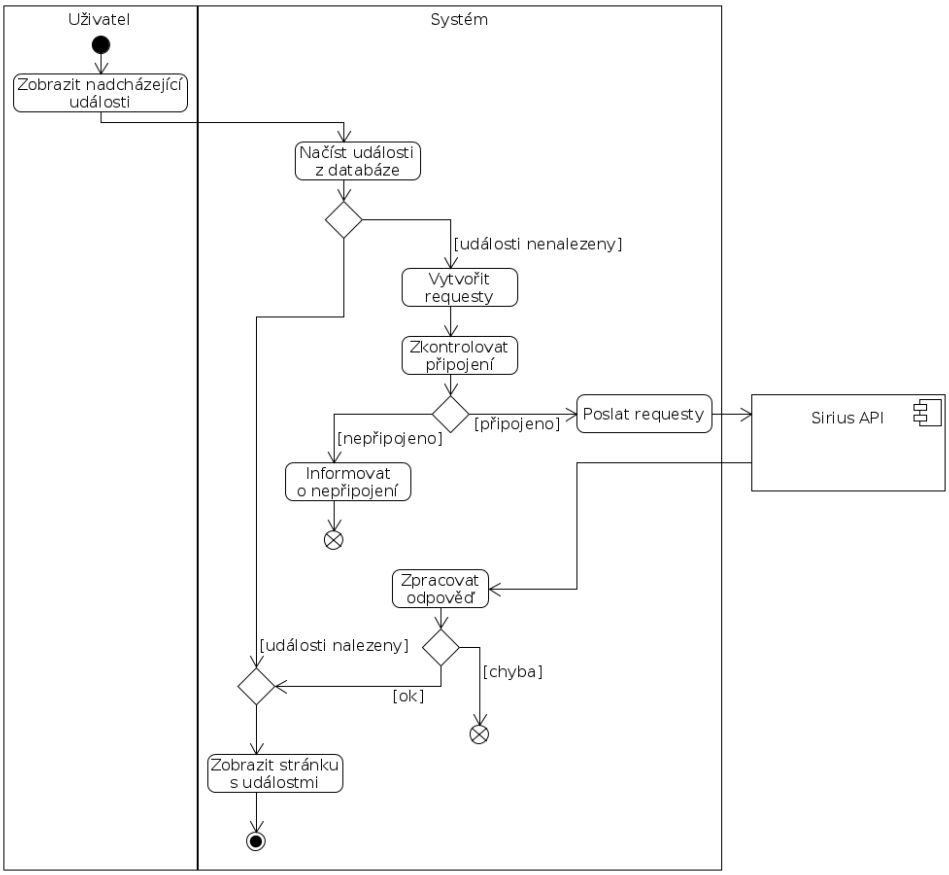


Obrázek 1.1: Use case diagram

po kterou uživatel chce, aby byla místnost volná.

Hlavní scénář: Uživatel nemá chtěné hledání v rychlé volbě

1. Případ užití začíná v okamžiku, kdy uživatel je přihlášen do aplikace, nachází se na úvodní obrazovce a chce provést hledání volných učeben podle specifikací.
2. Uživatel zvolí možnost nového hledání.
3. Systém se přepne do další aktivity s hledáním a zobrazí formulář umožňující specifikovat hledání.
4. Uživatel vyplní potřebné údaje a stiskne tlačítko hledat.
5. Systém získá informace ze serveru.
6. Systém zobrazí uživateli seznam místností, které odpovídají jeho požadavkům.
7. Systém uloží provedené hledání do rychlé volby.



Obrázek 1.2: Aktivity diagram zobrazení nadcházejících událostí

Alternativní scénář: Uživatel má chtěné hledání v rychlé volbě

1. Případ užití začíná ve 2. kroku hlavního scénáře.
2. Uživatel zvolí příslušné hledání v rychlé volbě.
3. Systém načte data z databáze.
4. Systém zobrazí uživateli seznam místností, které odpovídají jeho požadavkům.

UC4 Zobrazit rozvrhové události místnosti
Umožňuje zobrazení rozvrhových událostí místnosti.

UC5 Zobrazit obsazenost učeben
Umožňuje zobrazení seznamu učeben, kde každá graficky indikuje, zdali je

obsazená.

UC6 Přidat/Odebrat místnost
Umožňuje uživateli přidat popř. odebrat místnost do popř. ze seznamu obsazenosti místností.

Hlavní scénář: Uživatel přidá novou místnost do seznamu místností

1. Případ užití začíná v okamžiku, kdy uživatel je přihlášen do aplikace, nachází se na úvodní obrazovce.
2. Uživatel klikne na pole s volnou místností.
3. Systém se přepne do aktivity s místnostmi, která ukazuje obsazenost u jednotlivých místností
4. Uživatel klikne na tlačítko „+“ v action baru
5. Systém zobrazí dialog s textovým polem
6. Uživatel napíše kód učebny a potvrdí
7. Systém získá informace o učebně online
8. Systém přidá místnost k ostatním v aktivitě a uloží ji

UC7 Filtrovat události
Umožňuje uživateli filtrovat události podle vybraného typu události.

UC8 Filtrovat místnosti
Umožňuje uživateli filtrovat místnosti podle vybraného typu místnosti.

UC9 Vyhledat za pomoci rychlé volby
Umožňuje uživateli zobrazit výsledek již hledané informace, která je uložená v rychlé volbě.

UC10 Změnit připnutý stav hledání v rychlé volbě
Umožňuje uživateli změnit připnutý stav hledání v rychlé volbě na připnutý nebo nepřipnutý. Tento stav následně ovlivňuje, jestli se staré hledání v rychlé volbě přepíše hledáním novým.

UC11 Odstranit hledáni z rychlé volby
Umožňuje uživateli odstranit hledání z rychlé volby.

UC12 Aktualizovat rozvrhové události
Umožňuje uživateli aktualizovat jeho nadcházející rozvrhové událostí a události

místností.

UC13 Aktualizovat seznam učitelů a studentů
Umožňuje aktualizaci jmen, která se vyskytují na fakultě, a která slouží k našeptávání.

UC14 Přepnout uživatele
Umožňuje odhlášení přihlášeného uživatele, aby se do aplikace mohla přihlásit jiná osoba.

1.4 Doménový model

Doménový model (viz obrázek 1.3) představuje strukturovaný pohled na problémovou oblast. Popisuje entity, jejich atributy a vztahy mezi nimi.

1.4.1 Person

Entita Person uchovává informace o lidech na ČVUT. Uchovává atributy, jako jsou uživatelské jméno, které je unikátní v rámci celého ČVUT, celé jméno i s tituly a email.

1.4.2 Room

Entita Room definuje místnost v rámci ČVUT. Obsahuje informace o své kapacitě, typu a kódu učebny, který je unikátní.

1.4.3 ScheduleEvent

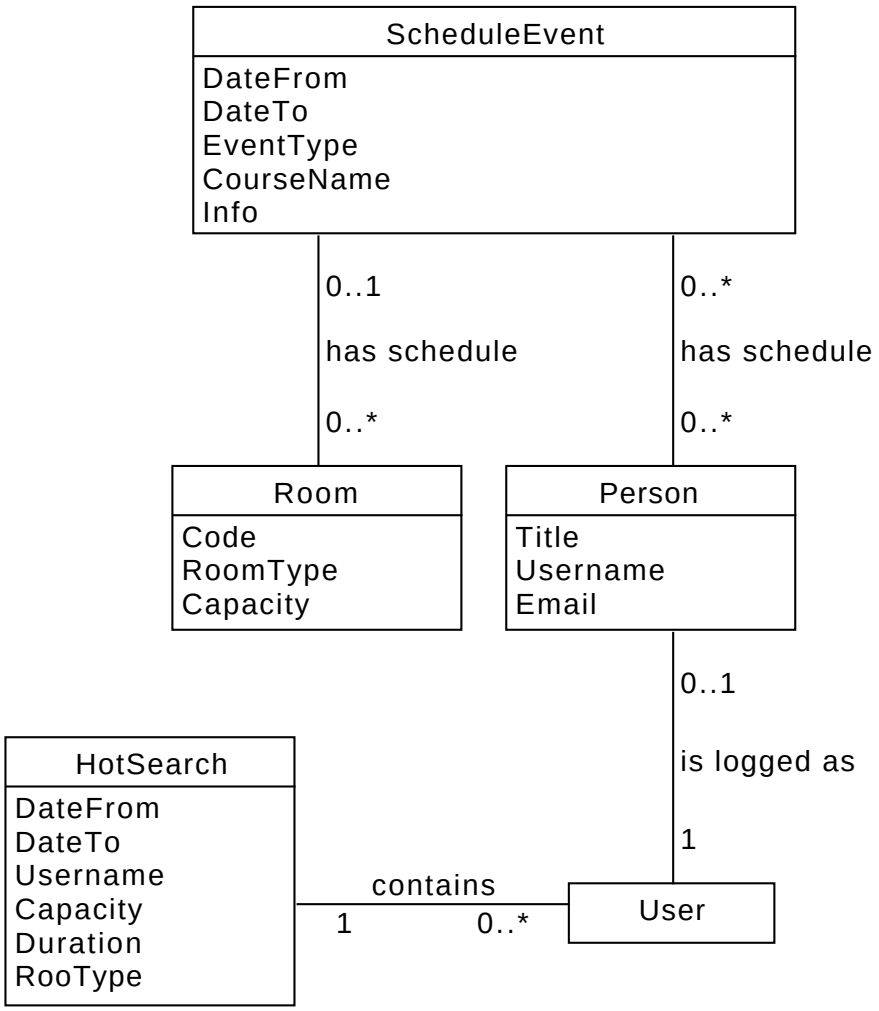
Entita ScheduleEvent je základním kamenem celé aplikace. Je to entita, která popisuje rozvrhovou událost. Má atributy, které definují, odkdy a dokdy událost probíhá a jakého je událost typu. Následně obsahuje název předmětu, pod kterým se událost vyskytuje a doplňující informace.

1.4.4 HotSearch

Entita HotSearch uchovává hledání, které uskutečnil uživatel. Ukládá v sobě čas vymezující hledání, hledané uživatelské jméno nebo informace potřebné k vyhledání volné místnosti, čímž je kapacita, typ místnosti a potřebný čas, po který má být místnost volná.

1.4.5 User

Entita User obsahuje informace o přihlašovacím klíči, jenž se uživateli přidělí po přihlášení. O tomto klíči se bude psát v sekci o autorizaci 3.1.



Obrázek 1.3: Doménový model

Kapitola 2

Návrh řešení

V návrhu řešení budu postupně popisovat možnosti řešení problémů, které připadaly v úvahu, posléze představím architekturu systému, kterou budu využívat a nastíním model databáze. Nakonec představím uživatelské rozhraní jednotlivých obrazovek, které bude aplikace poskytovat.

2.1 Možnosti řešení

Od výsledného řešení očekáváme, že splní všechny vytyčené cíle a funkce. Zejména je potřeba, aby řešení bylo jednoduché a intuitivní a aby se uživatel dostal k vyžadovaným informacím co možná nejrychleji a nic ho zbytečně nezdržovalo. Ze zadání plyne povinnost využít dostupných prostředků na ČVUT, čímž jsou výše zmíněné aplikační rozhraní Sirius a KOSapi.

2.1.1 Desktopová aplikace

Prvním možným řešením je desktopová aplikace. Aplikace by si spravovala vlastní databázi, kde by ukládala potřebné informace, které by stahovala z aplikačních rozhraní. Výhodou tohoto řešení by byl přístup k informacím, i kdyby počítač nebyl zrovna připojen k internetu. Nevýhodou je, že při potřebě získat informaci v co nejkratší době, je zapotřebí vytáhnout počítač, uvést ho do provozu a teprve poté se k hledané informaci dostaneme.

2.1.2 Aplikace v prohlížeči

Aplikace v prohlížeči je dalším možným řešením, které se místo přímo do počítače instaluje do webového prohlížeče. Aplikace by poskytovala informace i offline a při připojení na internet, by se sama aktualizovala. Výhodu má stejnou jako u aplikace desktopové, avšak má jednu povinnost navíc. K tomu, aby uživatel mohl k aplikaci přistupovat, musí mít nainstalovaný podporovaný prohlížeč.

2.1.3 Webová aplikace

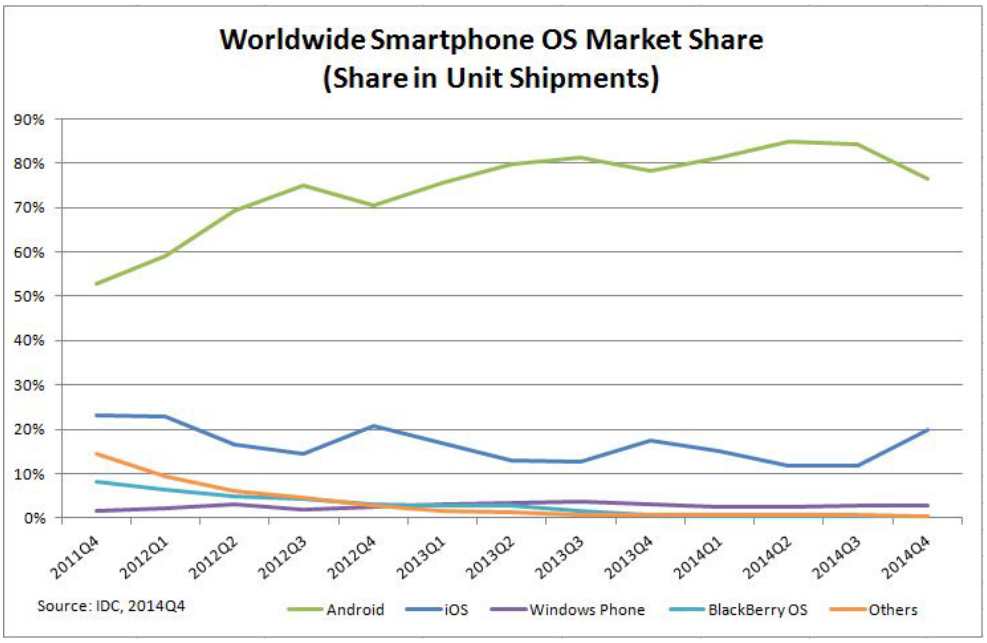
Webová aplikace oproti dvěma předchozím řešením funguje jenom s připojením na internet. Aplikace by běžela na serveru a kdokoliv by se na ní mohl připojit a zobrazit chtěná data. Webová aplikace by poskytovala podobné informace jako stránka Timetables, avšak informace by byly zaměřeny na rozvrhové události v aktuálním čase. Navíc by program měl jenom pár funkcí, což by způsobilo přehlednost a snazší použitelnost pro uživatele. Výhodou tohoto řešení je, že nikdo si nemusí nic instalovat a kdokoliv se na ní může připojit jak z PC, tak i z mobilního zařízení, což vyhledání informace značně urychlí a člověk nemusí kvůli tomu zapínat notebook. Zásadní nevýhodou tohoto řešení je, že bez připojení k internetu uživatel nezjistí nic.

2.1.4 Mobilní aplikace

Posledním řešením je mobilní aplikace. S mobilními aplikacemi se před nedávnou dobou roztrhl pytel a díra zůstává stále nezašitá. Mobilní aplikace se podobně jako aplikace desktopová nespoléhá pouze na internet, ale dokáže si ukládat informace do databáze a díky tomu poskytovat informace i offline. Nicméně oproti počítačové aplikaci, ta mobilní vede v rychlosti přístupu k datům v okamžicích, kdy záleží na každé vteřině a kdy upřednostníte vytáhnutí chytrého telefonu z kapsy místo vytahování PC z batohu. Nevýhodou mobilní aplikace je to, že má vícero mobilních OS, které jsou různě rozprostřené mezi uživateli, a tudíž je celkem obtížné a časově náročné je pokrýt všechny.

2.2 Zvolené řešení

Z možností jsem si vybral mobilní aplikaci. Je to především díky své snadné mobilitě a možnosti poskytovat data offline. S mobilní aplikací přichází důležité rozhodnutí týkající se operačního systému, na který bude aplikace stavěna. V úvahu jsem vzal nejrozšířenější tři operační systémy, které v současné době určují směr mobilních OS a těmi jsou Android (Google), iOS (Apple) a Windows Phone (Microsoft). Hlavním kritériem mé volby byl počet uživatelů, kterému by mohla být aplikace přístupná a to z celosvětových statistik[13] s přehledem vyhrává operační systém Android s více jak 70% podílem na trhu a více jak miliardou zařízení. Pro porovnání podílu na trhu s dalšími operačními systémy je rozdíl vyobrazen na obrázku 2.1. Dalším neméně důležitým kritériem je osobní zalíbení v Androidu a celkové sympatie ke společnosti Google. Proto jsem si operační systém Android vybral jako systém, na kterém budu stavět svou bakalářskou práci.



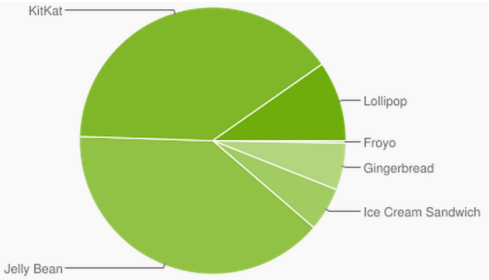
Obrázek 2.1: Celosvětový podíl na trhu mobilních operačních systému

2.2.1 Výběr platformy

V poslední řadě je ještě nutné specifikovat verzi platformy, na kterou budu aplikaci stavět. S každou verzí, kterou vývojáři Androidu vytvoří, přichází nové funkce a změny v uživatelském rozhraní. Těmito změnami však nijak neovlivňují již vydané aplikace díky tzv. zpětné kompatibilitě, která u Androidu funguje. Ta zajišťuje, že každá aplikace, která používá androidí funkce ze starších verzí, bude fungovat i na zařízeních s novou verzí. Nicméně tento systém způsobuje mnoho komplikací a vývojářům velmi znepříjemňuje život. Tím důvodem je to, že jestliže chtějí vytvořit aplikaci, která bude správně fungovat i na zařízeních se starší verzí Androidu, musí se podle toho zařídit. Na starších verzích totiž není ta nová úžasná funkce, kterou teď představili v nové verzi, a pakliže chceme onu super funkci využít, musíme si ji sami vytvořit. Existují však podpůrné balíčky od Android vývojářů, které to dělají za nás, avšak nejsou u všeho.

Ze statistik[14] viditelných na obrázku 2.2 můžeme vyčíst nejrozšířenější verze mezi uživateli chytrých zařízení s Androidem. Verze s distribucí menší než 0,1% nebo s verzí nižší než 2.2 tabulka neobsahuje. Poslední verzi, kterou budu podporovat je verze 4.0, která k aktuálnímu datu spolu s novějšími verzemi pokrývá více jak 90% zařízení s Androidem. Verzi 4.0 jsem si vybral proto, že přináší jednotné uživatelské rozhraní pro mobily a tablety a nové prvky a rozhraní z verze 3.x.

Version	Codename	API	Distribution
2.2	Froyo	8	0.3%
2.3.3 - 2.3.7	Gingerbread	10	5.7%
4.0.3 - 4.0.4	Ice Cream Sandwich	15	5.3%
4.1.x	Jelly Bean	16	15.6%
4.2.x		17	18.1%
4.3		18	5.5%
4.4	KitKat	19	39.8%
5.0	Lollipop	21	9.0%
5.1		22	0.7%



Obrázek 2.2: Distribuce verzí Androidu

2.3 Architektura systému

Aplikace bude řešena přes softwarovou architekturu Model-view-controller[15], která rozděluje systém do tří nezávislých částí. Pakliže by se jedna z komponent modifikovala, zbylé části by se musely upravit jen minimálně. V našem případě však budeme mít ještě jednu část a to část serverovou, ze které budeme čerpat aktuální data.

View

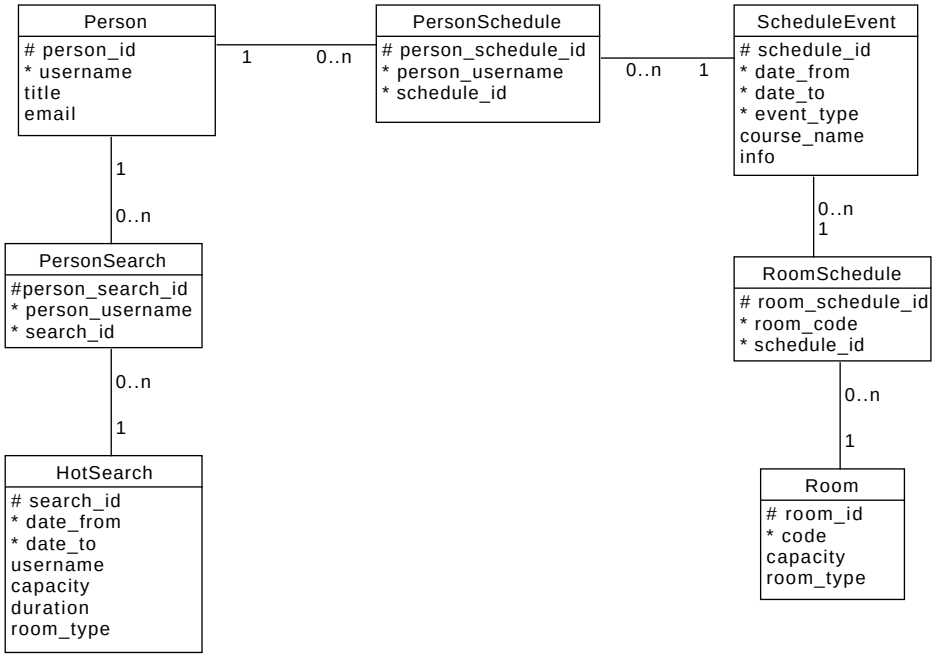
Komponenta View představuje uživatelské rozhraní. Při vyvíjení aplikace pro operační systém Android si můžeme komponentu View představit jako jednotlivé tzv. layouty. Layout je uložen ve formátu XML a reprezentuje vzhled celé obrazovky nebo na druhé straně vzhled jednoho řádku v seznamu. Upřesňuje jaké grafické prvky vzhled obsahuje a jakým způsobem jsou jednotlivé prvky uspořádány.

Model

Modelová část v sobě obsahuje veškerou logiku a výpočty. Poskytuje rozhraní, které zpřístupňuje data uložená v databázi. Model také komunikuje se serverovou částí, odkud stahuje data.

Controller

Controller zastupuje roli prostředníka mezi komponentami Model a View a propojuje je. Na jednu stranu plní jednotlivé grafické elementy daty, které si vytáhl z modelu a na druhou stranu data dává modelu, ať si je uloží. V rámci Androidu, komponentu Controller můžeme vidět v jednotlivých adaptérech, které fungují jako most mezi daty a grafickým elementem. Dalšími příklady Controlleru jsou listenery grafických elementů, které nějakým způsobem reagují na akci uživatele. V poslední řadě můžeme ještě zmínit třídu Activity. Třída



Obrázek 2.3: Databázový model

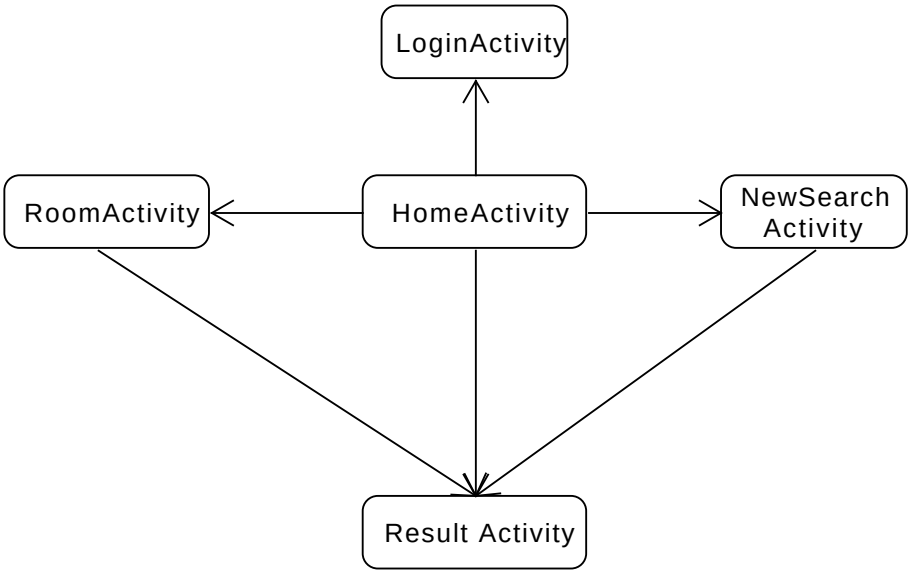
Activity představuje celou obrazovku. Přísluší jí přidělený layout a sama spravuje plnění grafických prvků daty.

2.4 Databázový model

K tomu, abychom mohli stahovaná data ukládat přímo do telefonu, nám bude sloužit databáze. Schéma databázové struktury je zachyceno na diagramu 2.3. Tabulky PersonSearch, PersonSchedule a RoomSchedule slouží k rozbití vztahů Many-To-Many a k lepšímu přístupu k datům. Zbývající entity vyplynuly z doménového modelu. Entita user zmíněná v doménovém modelu se bude s informacemi ohledně přihlašovacího klíče, ukládat do paměti telefonu do tzv. Shared preferences, místo do databáze. Je to z důvodu, že aplikace bude mít v jednom okamžiku uložený maximálně jeden klíč zrovna přihlášeného uživatele. Pokud dojde k přepnutí uživatele, klíč se přepíše novým.

2.5 Návrh uživatelského rozhraní

V této podkapitole bych chtěl nastínit vzhled, jak budou vypadat jednotlivé obrazovky. Obrazovky se budou mezi sebou přepínat tak, jak je to vyobrazeno na diagramu přechodu aktivit 2.4. Na některých obrazovkách se bude vysky-



Obrázek 2.4: Diagram přechodů aktivit

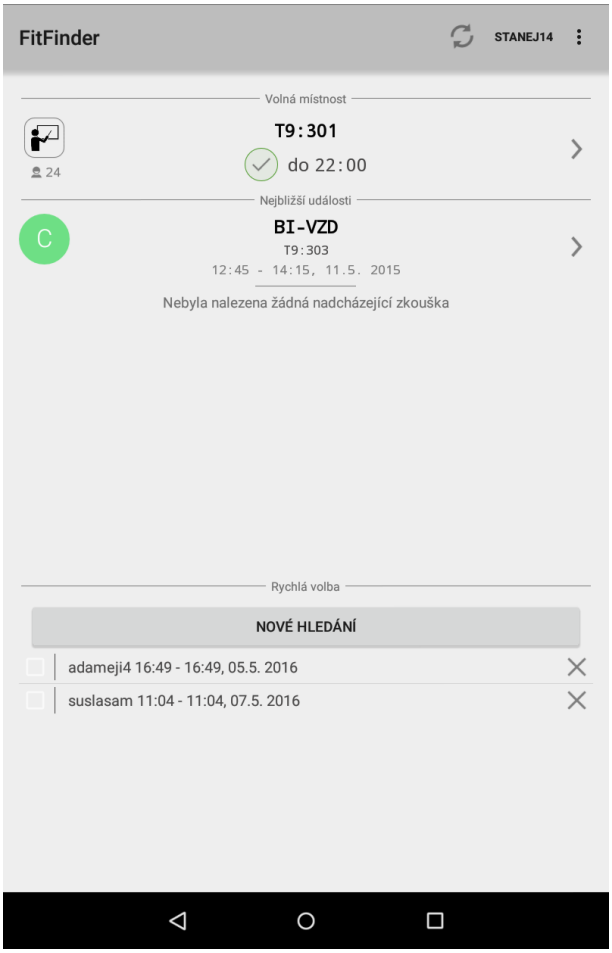
tovat tzv. Action Bar, což je horizontální menu, které se nachází na vrchu obrazovky. Action Bar se skládá z názvu a libovolného množství akcí, které jsou vyobrazeny ikonou nebo textem. Jako akci si můžeme představit aktualizaci nějaké informace či odhlášení uživatele.

2.5.1 Hlavní obrazovka

Hlavní obrazovka se skládá ze 3 částí a Action Baru. O každé části se posléze rozepíši jednotlivě. Jak můžeme vidět na obrázku2.5, jednotlivé části jsou opticky odděleny horizontální čarou, která má vždy uprostřed popisek označující, co následující sekce představuje. Action Bar obsahuje tlačítko na aktualizaci rozvrhových událostí, dále menu, které v sobě má možnost přepnout uživatele, a nakonec skrytou možnost aktualizace databáze jmen sloužící k našeptávání.

2.5.1.1 Volná místnost

Tato sekce zahrnuje informace o volné místnosti ze seznamu místností, která je aktuálně volná po nejdelší dobu. Po kliknutí na sekci nás aplikace přepne na obrazovku obsazenosti místností.



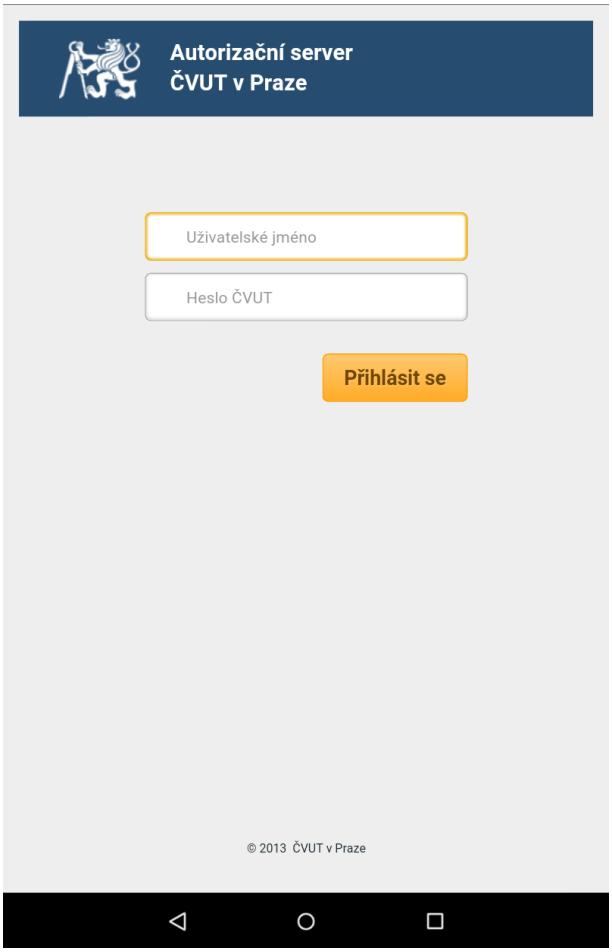
Obrázek 2.5: Hlavní obrazovka

2.5.1.2 Nejbližší události

V této části můžeme najít informace o nejbližší hodině a zkoušce, která uživatele čeká. Po kliknutí na sekci se zobrazí obrazovka se seznamem událostí uživatele.

2.5.1.3 Rychlá volba

Tato sekce obsahuje seznam uživatelem učiněných hledání, které může uživatel zopakovat pouhým kliknutím na konkrétní řádek. Každý řádek v sobě má tlačítko, které připne hledání, aby ho žádné nové nenahradilo a tlačítko, které daný řádek smaže. Pokud uživatel klikne na řádek mimo tlačítka, aplikace provede hledání a přepne se na obrazovku se seznamem událostí provedeného hledání. Nakonec je zde tlačítko „Nové hledání“, které nás přesune na obrazovku s novým hledáním.



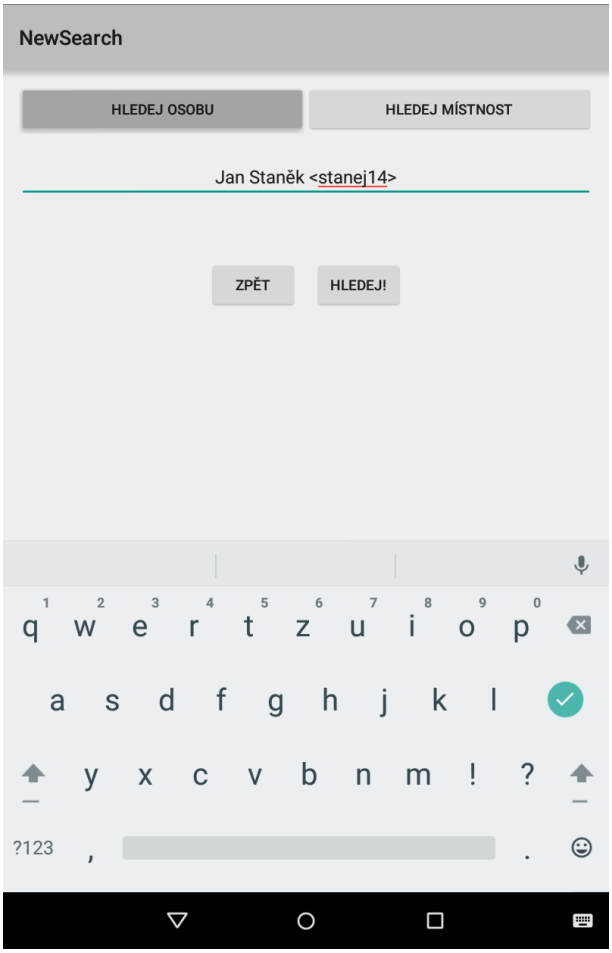
Obrázek 2.6: Přihlášení

2.5.2 Přihlášení

Přihlašovací obrazovka je to první, co uživatel uvidí při prvním spuštění aplikace. Přihlašovací stránku 2.6 je nutné vyplnit, aby aplikace mohla čerpat informace pod uživatelovou identitou. Po přihlášení se uživatel dostane k této přihlašovací obrazovce, pokud zvolí možnost přepnout uživatele na hlavní obrazovce. Na obrázku můžeme vidět webovou stránku, kterou nám již takovou dodává přímo stránka s autorizací. V příslušných kolonkách vyplní uživatel svůj login a dá přihlásit. Pokud vše proběhne v pořádku, aplikace se přepne na hlavní obrazovku.

2.5.3 Nové hledání

Obrazovka2.7 se skládá ze dvou částí, kde každá má společné 2 tlačítka a to jsou tlačítko „Hledej!“, které provede specifikované hledání a tlačítko „zpět“,



Obrázek 2.7: Nové hledání

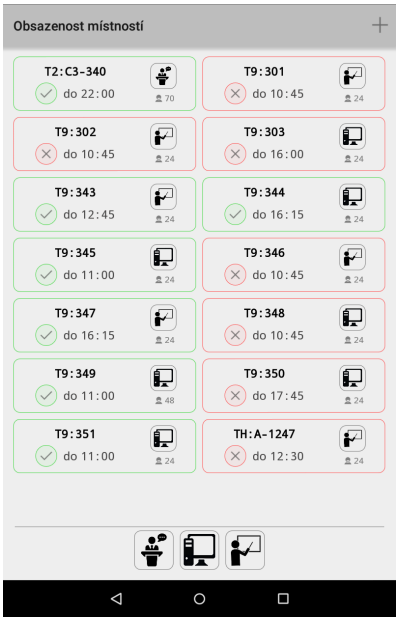
které nás vrátí na hlavní obrazovku. Mezi těmito částmi se přepíná na vrchu obrazovky pomocí tlačítek „Hledej místnost“ a „Hledej osobu“.

2.5.3.1 Hledání volné místnosti

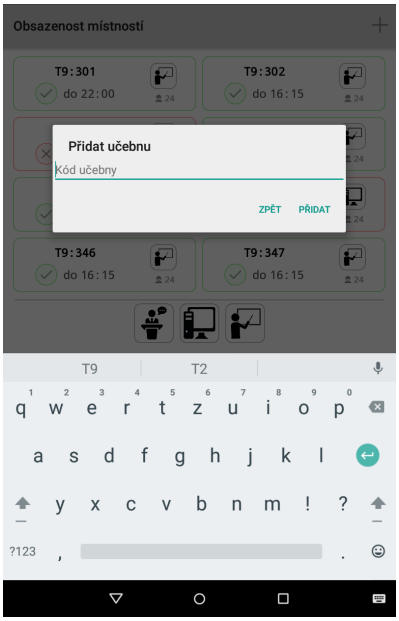
Tato část nám umožní specifikovat parametry, které od volné místnosti požadujeme. Můžeme specifikovat její typ, kapacitu a dobu po jakou chceme, aby byla místnost volná.

2.5.3.2 Hledání osoby

Při hledání osoby se nám zobrazí pouze jedno pole, kam uživatel bude za pomoci našeptávání vyplňovat jméno hledaného nebo jeho uživatelské jméno.



Obrázek 2.8: Přehled místností



Obrázek 2.9: Přidání místností

2.5.4 Obsazenost místností

Na obrazovce 2.8 můžeme vidět seznam místností, kde u každé místnosti je zobrazen její kód, kapacita a typ místnosti. Dále vidíme ikonku, která indikuje, zdali je místnost obsazená nebo volná a čas dokdy tento stav bude trvat. Rámeček kolem místnosti má také v závislosti na obsazenosti odlišnou barvu. Po kliknutí na rámeček se aplikace přepne na obrazovku se seznamem událostí dané místnosti. Pod oddělující čárou vidíme tlačítka, která slouží k filtrování místností podle jejich typu. Po kliknutí na tlačítko s filtrováním se zobrazí pouze ty místnosti, které odpovídají ikoně. Po opětovném kliknutí na stejné tlačítko, se filtrování vypne a zobrazí se opět místnosti všechny. Přidat místnost do seznamu se dá po kliknutí na tlačítko „+“ na pravé straně Action Baru. Po kliknutí na něj se zobrazí dialogové okno 2.9, kde aplikace uživatele vyzve k zadání kódu učebny. Místnost se dá ze seznamu odebrat po kliknutí na možnost „Odebrat místnost“, která se objeví v okamžiku, kdy uživatel podrží prst na rámečku s místností.

2.5.5 Seznam událostí

Na této obrazovce zobrazeného na snímku 2.10 zaujímá většinu plochy seznam nadcházejících rozvrhových událostí subjektu, který je zmíněn v názvu Action Baru. U každé události můžeme vidět informace týkající se kódu předmětu, místnosti, kde se událost koná a času. Na levé straně každého řádku můžeme vidět symbol označující typ události. Všechny symboly můžeme vidět ve spodní

Obrázek 2.10: Seznam událostí

části obrazovky, kde slouží k filtrování událostí. Filtrování funguje stejně jako u seznamu místností. Po kliku na typ událostí se zobrazí pouze ty události, které mají stejný symbol, a jsou tudíž stejného typu. Po opětovném kliknutí se filtrování vypne. Obrazovka již nenabízí žádnou možnost přejití na jinou obrazovku.

Kapitola 3

Realizace

V této kapitole bych chtěl popsat zajímavé problémy, se kterými jsem se v průběhu implementace projektu setkal a popsal, jak jsem je řešil.

3.1 Autorizace

Autorizace je v každé aplikaci jedna z nejvíce zapeklitých věcí, která dělá samotným uživatelům vrásky na čele. Uveďme jako příklad aplikaci, která představuje emailový komunikátor na vaši oblíbené internetové stránce a která potřebuje vaše přístupové údaje k tomu, aby mohla správně fungovat. Každý chvilku váhá, zda informace poskytnout, nebo raději využívat emailový komunikátor přímo na stránce. Většinou se objeví otázky, jestli si aplikace nějakým způsobem přihlašovací údaje uchovává, nebo je rovnou maže či jestli se někdo, kdo má můj telefon může k mým přihlašovacím údajům nějak dostat. Autorizací potom aplikace zjistí, kdo ji zrovna používá a může se uživateli přizpůsobit.

V aplikaci potřebujeme zjistit, kdo telefon drží v ruce, abychom uživateli mohli poskytnout jeho nejbližší hodiny a jeho nejbližší zkoušky. Navíc je třeba se dotazovat aplikačních rozhraní (KOSapi, Sirius) pod identitou přihlášeného, jelikož jsou zabezpečené a potřebují ověřit toho, kdo se přihlašuje.

3.1.1 AppsManager

Autorizace k aplikačním rozhraním ČVUT jako jsou KOSapi a v budoucnu i Sirius API je vyřešena elegantním způsobem a v podstatě vývojáři aplikací již mají vydlážděnou cestu. Jediné, co je třeba udělat je zaregistrovat si aplikaci přes Apps Manager[16], který je pod správou ČVUT. Tam si vytvoříme projekt a vybereme si, které aplikační rozhraní chceme využívat. Následně dostaneme přidělené client_id, které slouží jako přihlašovací jméno aplikace. Nyní stačí uživatele přesměrovat na stránku s přihlášením do Apps Manageru, který nám

po úspěšné autorizaci poskytne access token, který slouží jako klíč do API, které chceme využívat. Tento způsob autorizace využívá OAuth standardu.

3.1.2 OAuth 2.0

OAuth je protokol, který poskytuje bezpečnou autorizaci z webových, mobilních a desktopových aplikací. Hlavní výhodou tohoto standardu je, že uživatel může poskytnout přístup ke svým datům službě, aniž by službě jakýmkoliv způsobem zadával přihlašovací údaje[17].

Celá autorizace jednoduše proběhne tak, že aplikace přesměruje uživatele na stránku s přihlášením do Apps Manageru, kde ho vyzve, aby se přihlásil. Do url stránky ještě přidá parametr, ve kterém specifikuje své id, které dostal při registraci. Díky tomu uživatel uvidí, jaké aplikaci a k jakým datům poskytuje přístup. Tento proces probíhá mimo aplikaci, a tudíž není způsob, jak se dostat k přihlašovacím údajům. Po úspěšném přihlášení se stránka přesměruje na námi zvolenou adresu, která bude mít v url přidané informace o access tokenu, který byl uživateli přidělen. Kromě samotného tokenu, je mezi informacemi obsažen datum, do kterého token platí, a oblasti, kam má aplikace s tokenem přístup.

Nyní stačí přidat k dotazu na aplikační rozhraní ČVUT přidělený access token, který nám požadované API odemkne.

Abychom zjistili, komu token patří, můžeme přejít na adresu, která poskytuje informace o tokenu. Tam v našem případě získáme username uživatele, jeho email, datum trvanlivosti tokenu a další.

Co se týče přístupu na Sirius API, tak jeho autoři ještě OAuth neimplementovali, avšak mají to v plánu. Místo přihlašování se, Sirius vygenerované tokeny rozeslal uživatelům na email. Ne každý tento email zaregistroval, a navíc bylo by zdlouhavé, kdyby každý uživatel musel svůj access token přeťukávat. Proto prozatím v aplikaci budeme používat univerzální token, který má přístup ke všem rozvrhům.

3.2 Získávání dat

Jak jsem již zmínil výše, primárním zdrojem, ze kterého budu získávat informace je API, které bylo vytvořeno pod názvem Projektem Sirius. Sirius mi poskytne aktuální rozvrhová data, která budou zohledňovat veškeré plánované i neplánované rozvrhové výjimky. Nicméně je ještě nutné získat data o studentech, učitelích a o učebnách na FIT. K tomu budu využívat informační systém KOS, na který budu přistupovat přes aplikační rozhraní KOSapi.

3.2.1 Sirius

Sirius poskytuje chtěná rozvrhová data ve formátu JSON a iCalendar. Pro naši aplikaci je vhodnější data získávat ve formátu JSON, jelikož Android sám

o sobě poskytuje pomocné třídy na snadnější pracování s formátem JSON. K datům se můžeme dostat pod requesty typu GET. Každý tento dotaz (request) může následovat skupina url parametrů, které se píší za otazník, který přidáme za samotnou url adresu a vzájemně se oddělují symbolem ampersand („&“). Nám přístupné parametry specifikují, v jakém časovém rozmezí chceme, aby se rozvrhové události nacházely, a kolik jich chceme zobrazit na stránku. Zmíněné parametry jsou nepovinné, avšak každý request musí mít v sobě parametr s klíčem přihlášeného uživatele. Nepovinné parametry jsou následující:

- limit – Upravuje kolik výsledků zobrazit, limit je omezen shora 1000; výchozí hodnota je 20
- offset – Upravuje, o kolik výsledků se má posunout dopředu, a tudíž je vynechat; výchozí hodnota je 0
- from – Upravuje levou mez na časovém horizontu
- to – Upravuje pravou mez na časovém horizontu

Ačkoliv má Sirius více adres, kam můžeme přistupovat přes GET requesty, my využijeme jen následujících dvou.

/people/{username}/events
Pod tímto requestem, při dosazení uživatelského jména za „{username}“, nám SiriusApi vrátí rozvrhové události dané osoby.

/rooms/{kosId}/events
Pod tímto requestem, při dosazení platného kódu místnosti (T9:301) za „{kosId}“, nám SiriusApi vrátí rozvrhové události dané místnosti.

U obou dotazů nám server vrátí stejně strukturovaný JSON v odpověď. Z něj můžeme vyčíst potřebné informace. Těmi jsou čas, kdy událost trvá, typ události a název kurzu. Dále potřebujeme znát data, která jsou uložená pod označením meta, což jsou meta informace o odpovědi, kterou nám server vrátil a která nám slouží k orientaci mezi výsledky. Má v sobě informaci ohledně počtu zobrazených výsledků. Pakliže se tento počet rovná limitu, musíme zavolat tentýž GET request s posunutým parametrem offset o tolik, kolik je aktuální limit. Tím budeme mít jistotu, že získáme všechny události, které se pod zadaným požadavkem vyskytují.

3.2.2 KOSapi

Aplikační rozhraní KOSapi zprostředkovává část dat z databáze KOS, týkající se bílé knihy, rozvrhů, studentů, učitelů, místností a dalších. Na rozdíl od Siria

KOSapi poskytuje data ve formátu Atom, což představuje novou generaci formátů založených na XML[5]. Je to formát, který podporuje syndikaci obsahu, což znamená, že podporuje aktuálnost dat mezi aplikacemi[6]. K datům se opět dostaneme pod requesty typu GET.

KOSapi disponuje četnými adresami, které dohromady poskytují všechny potřebné informace týkající se vesměs všeho oficiálního na fakultě. My si avšak vystačíme pouze se třemi adresami, které nám poskytnou data lidí a místností na fakultě. Podobně jako je to u API Projektu Sirius, KOSapi má také svůj set url parametrů, kterými můžeme filtrovat výsledky. Kromě již zmíněných parametrů limit a offset KOSapi nabízí mnohé další, my však budeme používat pouze jeden navíc a tím je parametr query, který umožní vyhledávat mezi záznamy podle atributů. Parametr query má podobu výrazu napsaného v dotazovaném jazyce RSQL, který vyvinul Jakub Jirůtka.

Příkladem můžeme uvést výraz:

```
query=faculty="18000";studyState=="ACTIVE".
```

Výraz vymezuje ty výsledky, které mají atribut studyState rovný řetězci ACTIVE a které mají atribut division rovný řetězci 18000. Při přístupu na KOSapi opět využijeme klíč přihlášeného uživatele.

Adresy, které aplikace využívá, jsou:

/students

Pod tímto requestem KOSapi vypíše všechny studenty, kteří se vyskytují na ČVUT. Z výpisu nás posléze zajímají pouze záznamy title, username a email.

/divisions/18000/teachers

Pod tímto requestem získáme učitele, kteří jsou spjatí s Fakultou informačních technologií. Proto se dotazujeme na divizi číslo 18000, jelikož ta tuto fakultu zastupuje. Z výpisu nás zajímají stejné položky jako u studenta.

/rooms

Jako poslední na řadě je potřeba získat informace o místnostech, které získáme na adrese /rooms. Z výpisu si posléze vytáhneme údaje o názvu učebny, její kapacitě a jakého je učebna typu. Jako příklad je níže uvedena část odpovědi, jak vypadá zjednodušeně záznam místnosti s kódovým označením T9:301.

```
<atom:entry>
    <atom:title>T9:301</atom:title>
    <atom:id>urn:cvut:kos:room:1074106</atom:id>
    <atom:updated>2011-11-28T16:23:03.0</atom:updated>
```

```
    <atom:author>
      <atom:name>halaska</atom:name>
    </atom:author>
    <atom:link rel="self" href="rooms/T9:301/"/>
    <atom:content>
      <capacity for=\"examining\">24</capacity>
      <capacity for=\"teaching\">24</capacity>
      <code>T9:301</code>
      <division>Fakulta informacnich technologii</division>
      <locality>DEJ</locality>
      <name>NBFIT ucebna</name>
      <type>TUTORIAL</type>
    </atom:content>
</atom:entry>
```

3.3 Aktualizace dat

Dalším zajímavým problémem je aktualizace informací, které se týkají jak rozvrhových událostí, tak informací ohledně lidí a místností na FIT. Zapeklitost je v tom, kdy data aktualizovat a jestli je obnovovat automaticky bez přičinění uživatele, nebo naopak čekat, až si je uživatel obnoví sám. V podstatě jde o souboj mezi dvěma prioritami a těmi jsou aktuálnost a šetření internetového připojení.

Aktualizovat jména studentů a učitelů na fakultě je operace celkem zdlouhavá a zabere se stabilním připojením několik vteřin, než se data stáhnou. Proto se tento proces spouští jen v případě, kdy si to uživatel sám vyžádá kliknutím na tlačítko na hlavní obrazovce. Další operace, která se spouští tlačítkem je aktualizace rozvrhových událostí seznamu místností a událostí přihlášeného uživatele. Toto tlačítko je opět přístupné z hlavní obrazovky.

Automatická aktualizace rozvrhových událostí přichází na scénu v momentě, kdy se aplikace spustí nebo obnoví a v databázi není žádná nadcházející rozvrhová událost uživatele nebo místnosti. Za automatické stahovaní dat ještě můžeme označit stáhnutí informací o místnosti, kterou si uživatel sám přidá do seznamu nebo vyhledání rozvrhu někoho na FIT.

3.4 Tvorba uživatelského rozhraní

Tvorba uživatelského rozhraní je pro vývojáře vždy výzva. Musí vše udělat tak, aby se rozhraní ukazovalo bezproblémově na co nejvíce typů zařízení a verzí operačního systému. Můžeme mít zařízení, které má uhlopříčku pár centimetrů a stejné množství pixelů jako externí monitor a na druhé straně kolos, který se nedá pomalu držet v jedné ruce, ale rozlišení má mizerné. Dále vývojář musí mít neustále na paměti, že každý uživatel myslí trochu

jinak, a tudíž každý grafický element by měl co nejvíce sám o sobě na první pohled vypovědět, k čemu je dobrý a co dělá. Jsou uživatelé, kteří si nedokáží představit zadávat text do mobilního zařízení jinak než horizontálně a jsou tací, kteří když nedostanou odpověď na určité gesto do sekundy, začnou zběsile klikat na celou obrazovku ve snaze proces urychlit.

3.4.1 Rozpoznávání interaktivního elementu

Při vytváření uživatelského rozhraní jsem aplikaci často dával do rukou potencionálních budoucích uživatelů, aby si s aplikací pohráli. Často se dalo pozorovat, že uživatel s aplikací zápasí a na hlavní obrazovce hledá, na co se dá kliknout. Bylo tedy nutné přijít s prvky, které by uživateli naznačily, že tento element je klikatelný.

V prvé řadě se klikatelnému elementu přidala zpětná vazba, která způsobuje změnu pozadí a ohraničení elementu, pakliže se na něj klikne. Nicméně tato změna nešla bez interakce uživatele vidět, a tudíž nepřinesla kýžený výsledek. Nakonec prvek, který se ukázal jako efektivní, je objekt ve tvaru „>“ na kraji elementu. Tento objekt svým vzhledem indikuje, že se pod prvkem, ve kterém se nachází, něco skrývá a že se po kliknutí odkryje. Uživatelé Androidu jsou na něj zvyklí ze systémových aplikací, kde se vyskytuje otočený o 90° po směru popř. proti směru hodinových ručiček a kde představuje akci na odkrytí popř. zakrytí stahovacího menu.

3.4.2 Podporování více typu mobilních obrazovek

Jak již bylo zmíněno v úvodu sekce, na světě existují zařízení s různými typy obrazovek a s různými rozlišeními. Tudíž, abychom mohli pokrýt co možná nejvíce těchto typů, je zapotřebí tomu přizpůsobit i uživatelské rozhraní aplikace. Můžeme vytvořit ostrý obrázek ve tvaru čtverce o délce 200 pixelů. Pakliže tento obrázek zobrazíme na dvou rozdílných zařízeních, kde jedno bude mít rozlišení 480x800 pixelů a druhé 1440x2560 pixelů, tak na prvním uvidíme zřetelně celý obrázek a na druhém jakousi miniaturu čehosi nerozpoznatelného nebo roztáhnutý původní obrázek, u kterého také neurčíme, o co se jedná. Při vývoji Androidu se tento problém řeší vytvořením skupiny stejných obrázků s různým rozlišením, kde každý z nich reprezentuje obrázek, který je stavěný na určité rozlišení. Dalším způsobem je vytvoření obrázku typu .9.png, kterému se určí části, které se dají roztahovat a které mají zůstat nedotčeny a následně každému zařízení se přizpůsobí[18].

Další věcí je, že grafické elementy můžou vypadat na průměrném telefonu pěkně a strukturovaně. Pakliže je nepřizpůsobíme žádnému jinému rozlišení a zobrazíme si je na tabletu, elementy se roztáhnou a zbytečně zaberou prostor, kde by se mohl vyskytovat další prvek. Tento problém se opět řeší vytvořením skupiny layoutů (viz 2.3), kde tyto skupiny představují rozložení na určitý typ rozlišení. Tyto layouty se používají i při rotaci obrazovky.

Kapitola 4

Testování

V této kapitole popíši dva způsoby testování, kterými byla aplikaci podrobena. V první fázi proběhlo testování aplikace na různých mobilních zařízeních a posléze proběhlo testování uživateli.

4.1 Testování funkčnosti na rozdílných zařízeních

Testování probíhalo zejména na virtuálních zařízeních, které jsou poskytovány v rámci vývojového prostředí Android Studio od společnosti Google. Testovací zařízení se navzájem lišily ve velikosti obrazovky a hustotě pixelů. Dále měly rozdílné verze OS Android, kde nejnižší verze byla verze 4.0 a nejvyšší 5.1. Cílem testování bylo zejména najít nedokonalosti v zobrazování uživatelského rozhraní a popř. odhalení špatné funkčnosti.

Výsledkem testování bylo nalezení chybně určených velikostí elementů na obrazovkách s menší hustotou pixelů, což zapříčinilo přetékání grafických prvků přes sebe.

4.2 Testování uživatelské přívětivosti

Testování probíhalo vesměs na studentech FIT. Testy byly ve formě Usability testů, což jsou testy, které zkoumají chování uživatelů, když se jim poprvé představí aplikace. Testovacím subjektům se přidělí úkoly, které je nutné splnit a zadavatel pozoruje, jak se s každým problémem uživatel vypořádal a porovnává jeho akce s akcemi chtěnými[19]. Cílem testování bylo zjistit, zdali jsou všechny akce v aplikaci intuitivní a zdali uživatelům nedělá problémy splnit zadaný úkol. Po skončení posledního úkolu, byl testujícím poskytnut prostor na připomínky a náměty na zlepšení.

4.2.1 Testovací scénáře

Testovací scénáře definují úkol a předpokládané řešení, kterým se zadaný úkol plní. Scénáře byly následující:

1. **Zobrazte seznam nadcházejících událostí.**
   Řešení: Uživatel klikne na položku zobrazující jeho nejbližší událost na hlavní obrazovce.

2. **Vyhledejte rozvrh studenta s uživatelským jménem stanej14.**
   Řešení: Uživatel klikne na tlačítko „Nové hledání“ na hlavní obrazovce. Po přepnutí na obrazovku s novým hledáním vyplní uživatelské jméno do textového pole a stiskne tlačítko hledat.

3. **Zjistěte, která počítačová místnost je volná.**
   Řešení: Uživatel klikne na položku zobrazující volnou místnost na hlavní obrazovce. Po přepnutí na obrazovku se seznamem místností, uživatel klikne na ikonu počítačové učebny, čímž vyfiltruje ostatní učebny pryč ze seznamu. Následně podle zabarvení elementu představující místnost uživatel určí, která místnost je volná.

4. **Znovu vyhledejte studenta s uživatelským jménem stanej14.**
   Řešení: Uživatel klikne na položku představující již provedené hledání studenta stanej14 v rychlé volbě na hlavní obrazovce.

5. **Jaké události se budou konat v místnosti TH:A-1247.**
   Řešení: Uživatel klikne na položku zobrazující volnou místnost na hlavní obrazovce. Po přepnutí na obrazovku se seznamem místností, uživatel klikne na tlačítko "+", načež se zobrazí dialog na přidání místnosti, jelikož se hledaná místnost v seznamu nenachází. Uživatel vyplní jméno učebny a stiskne tlačítko „Přidat“. Následně klikne na nově zobrazenou místnost.

6. **Odstraňte místnost TH:A-1247 ze seznamu.**
   Řešení: Uživatel podrží prst na místnosti, dokud se mu nezobrazí nabídka na vymazání místnosti. Uživatel poté klikne na tlačítko „Vymazat“.

4.2.2 Výsledek testování

V průběhu testování se v jednotlivých úkonech většiny účastníků objevila prodleva z důvodu nedokonalé navigace mezi obrazovkami. Uživatelům zabralo pár vteřin na projetí všech informací zobrazených na hlavní stránce, než dospěli k závěru, které kliknutí je dovede na kýžený výsledek. Tato prodleva však postupně při plnění úkolů mizela.

Největší problém se projevil v úkolu, kdy měli účastníci zjistit, jaké události se budou konat v místnosti TH:A-1247. Uživatelé věděli, že tato místnost se

nenachází v seznamu místností, a tudíž někteří hledali odpověď na obrazovce s novým hledáním. Po neúspěšném hledání, kam by se dal zadat kód místnosti, uživatelé přešli na správnou obrazovku a objevili tlačítko „+“.

Menší prodleva se ukázala i v odebrání místnosti. Každý účastník se ihned vydal na správnou obrazovku a hledal tlačítko, které by indikovalo odstranění místnosti. Polovina účastníků našla odpověď skoro ihned, další po chvilce a jeden hledání vzdal.

Domnívám se, že tyto problémy byly vesměs způsobeny tím, že uživatelé viděli aplikaci poprvé a museli se zorientovat. Po sžití se s aplikací by měly tyto problémy vymizet. Řešením počáteční zmatenosti může být nápověda, která bude uživatele po prvním navštívení nějaké obrazovky informovat o funkcionalitách, které zde může provést.

Většinu účastníků také zajímalo, jak je nakládáno s přístupovými údaji, které jsou vyžadovány ihned po prvním spuštění aplikace. Věřím, že některé by tento mezikrok mohl odradit od používání této aplikace, pakliže nebudou znát proces autorizace. Proto tento proces bude v aplikaci popsán pod tlačítkem „?“, které se bude nacházet v Action baru na obrazovce s přihlášením.

Zde jsou náměty na zlepšení, které přišly od účastníků.

- Přidat informaci o poslední aktualizaci
- Možnost přihlásit se na zkoušky
- Možnost zobrazit společné hodiny s jiným studentem
- Možnost zobrazit, kdy probíhá paralelní hodina rozvrhové události

Závěr

Cílem této práce bylo vytvořit aplikaci, která by zobrazovala rozvrh uživatele i bez neustálého připojení k internetu a poskytovala možnost najít rozvrh jiných a zobrazit volné místnosti. Dále bylo cílem vytvořit příjemné intuitivní uživatelské rozhraní a v neposlední řadě aplikace měla využívat již vytvořené systémy na půdě fakulty.

Cíle považuji za vesměs splněné, ačkoliv je zde pár věcí, které se zatím nestihly dodělat. Je to z důvodu, že jsem kladl vyšší prioritu na návrh aplikace a na uživatelské rozhraní. Na zbytek pak již nezbyl čas. Jako zásadní nedokončenou věc považuji vyhledání učebny podle specifikací a nedodělané připínání rychlé volby. Dále není přístupné našeptávání místností a vymezení hledání rozvrhových událostí studenta popř. učitele podle času. Všechny tyto body se chystám doplnit v budoucnu.

Práce mi přinesla mnoho cenných zkušeností z oblasti implementace mobilní aplikace na OS Android, způsobu získávání dat z aplikačních rozhraní a také zkušenosti ze samotného návrhu. Doufám, že aplikace nebude přínosem jen pro mě, ale že se bude aktivně využívat mezi studenty a vyučujícími.

Budoucí práce

Vývoj aplikace považuji za dlouhodobý projekt, o který se budu chtít starat i v budoucnu a dostat ho k co nejvíce uživatelům. V blízké budoucnosti bych chtěl dodělat věci, které se nestihly v rámci bakalářské práce. Posléze bych rád aplikaci nahrál na portál Google Play, aby se mohla začít používat. Budu sledovat změny Projektu Sirius, které by měly v budoucnu přinést poskytování jednorázových akcí, položky info u zkoušek a autorizaci za pomocí Apps Manageru. Tyto změny poté promítnu do aplikace.

V rámci vzdálenější budoucnosti bych rád rozšířil aplikaci na operační systémy iOS a Windows Phone, pokud by se aplikace stala oblíbenou.

Literatura

[1] FIT CTU Prague: Timetables [online]. [Cit. 2015-4-4]. Dostupné z: https://timetables.fit.cvut.cz/

[2] FIT CTU Prague: STUDIJNÍ INFORMAČNÍ SYSTÉM (KOS) [online]. [Cit. 2015-4-4]. Dostupné z: https://kos.is.cvut.cz/

[3] Jirůtka, J.: *KOS API jako webová služba*. Bachelor thesis, CTU in Prague, Faculty of Electrical Engineering, 2010. Dostupné z: https://dip.felk.cvut.cz/browse/pdfcache/jirutjak_2010bach.pdf

[4] Google Apps for Education [online]. 2014, [Cit. 2015-4-4]. Dostupné z: https://ict.fit.cvut.cz/~web/current/web/ict/GoogleApps/index.shtml

[5] KOSapi [online]. 2011, [Cit. 2015-4-4]. Dostupné z: https://kosapi.fit.cvut.cz/projects/kosapi/wiki

[6] Hammersley, B.: What Is Atom [online]. 2005, [Cit. 2015-7-5]. Dostupné z: http://www.xml.com/lpt/a/1619

[7] Szolár, T.: Projekt Sirius [online]. 2013, [Cit. 2015-7-5]. Dostupné z: https://rozvoj.fit.cvut.cz/Sirius/WebHome

[8] Sirius: CTU Time management API [online]. 2014, [Cit. 2015-4-4]. Dostupné z: https://github.com/cvut/sirius

[9] Sedivy, J.: Už máte KOSeek? [online]. 2013, [Cit. 2015-4-4]. Dostupné z: http://otevrenainformatika.blogspot.cz/2013/09/uz-mate-koseek.html

[10] David, O.: Kosaak [Mobile application software]. 2015, [Cit. 2015-4-4]. Dostupné z: https://www.windowsphone.com/en-nz/store/app/kosaak/baf6d340-636b-400e-8e85-4aabbe9eac93

[11] O APLIKACI [online]. [Cit. 2015-7-5]. Dostupné z: https://navigator.fit.cvut.cz:4443/CVUT_Navigator/O_nas.html

[12] Arlow, J.; Neustadt, I.: *UML 2 a unifikovaný proces vývoje aplikací*. Brno: Computer Press, a.s., 2. aktualizované a doplněné vydání vydání, 2008, ISBN 978-80-251-1503-9.

[13] IDC Corporate USA: Smartphone OS Market Share, Q4 2014 [online]. 2015. Dostupné z: `http://www.idc.com/prodserv/smartphone-os-market-share.jsp`

[14] Google Inc.: Dashboards [online]. 2015. Dostupné z: `https://developer.android.com/about/dashboards/index.html`

[15] Pecinovský, R.: *Návrhové vzory: 33 vzorových postupů pro objektové programování*. Brno: Computer Press, a.s., 2007, ISBN 978-80-251-1582-4.

[16] CTU Prague 2014: O aplikácii [online]. [Cit. 2015-7-5]. Dostupné z: `https://auth.fit.cvut.cz/manager/about.xhtml`

[17] Jirůtka, J.: OAuth 2.0 [online]. 2014. Dostupné z: `https://rozvoj.fit.cvut.cz/Main/oauth2`

[18] Google Inc.: Draw 9-patch [online]. [Cit. 2015-11-5]. Dostupné z: `http://developer.android.com/tools/help/draw9patch.html`

[19] Stepp, M.: Usability Testing. [Cit. 2015-7-5]. Dostupné z: `https://courses.cs.washington.edu/courses/cse403/13sp/lectures/17-usabilitytesting.pdf`

Příloha **A**

Seznam použitých zkratek

API Application Programming Interface

CTU Czech technical university

ČVUT České vysoké učení technické

F Funkční požadavek

FEL Fakulta elektrotechnická

FIT Fakulta informačních technologií

JSON JavaScript Object Notation

N Nefunkční požadavek

OS Operační systém

REST REpresentational State Transfer

RSQL RESTful Service Query Language

SSO Single sign-on

UC Use case

UI User interface

URL Uniform Resource Locator

XML Extensible Markup Language

Příloha **B**

Obsah přiloženého CD

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