Západočeská univerzita v Plzni

Fakulta aplikovaných věd

Katedra informatiky a výpočetní techniky

Diplomová práce

Vizualizace fyziologických dat pro platformu Android

Plzeň, 2017 Renata Benešová

Prohlášení

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

V Plzni dne 28. 6. 2017

Renata Benešová

Poděkování

Ráda bych na tomto místě poděkovala vedoucímu práce Ing. Davidu Tolarovi za odborný dohled, konzultace k tématu práce, za pomoc a rady při zpracování této práce. Děkuji též výzkumnému centru Nové technologie Západočeské univerzity za možnost podílet se na tomto zajímavém výzkumném projektu.

Abstract

Measurement of body temperature can be monitored by the course of the disease and the effectiveness of the treatment of the patient is monitored. New Technologies Research Centre of University of West Bohemia is developing a wireless thermometer system that measures and records body temperature. The system has been tested at Klatovy's Hospital with result that the optimal way to access the measured temperature values is via the touch panel, which will be located in the head nurse room, or other suitable locations.

For the purpose of the measured data visualization, there has been previously developed a web application that was primarily used for the detailed analysis of the measured data. However, its practical use is not appropriate due to different functionality requirements of application. The goal of this thesis is development of Android application for the touch screen panels that access to patient temperature information and temperature sensors state.

Abstrakt

Měřením tělesné teploty může být monitorován průběh onemocnění a sledována účinnost léčby pacienta. Nové technologie – výzkumné centrum Západočeské Univerzity v Plzni v současnosti vyvíjí systém bezdrátových senzorů, které měří a zaznamenávají hodnoty tělesné teploty. Z testování systému, uskutečněného v Klatovské nemocnici, a.s., vyplynulo, že optimálním způsobem, jak zpřístupnit teploty pacientů personálu, je pomocí dotykového panelu, který bude umístěn v místnosti vrchní sestry, případně na dalších vhodných místech.

K vizualizaci naměřených dat byla již dříve vyvinuta webová aplikace, která sloužila primárně pro detailní rozbor naměřených dat. Její použití ale v praxi není vhodné z důvodu odlišných požadavků na funkčnost aplikace. Cílem tohoto projektu je vyvinout aplikaci pro dotykový panel s operačním systémem Android, která přehledně zpřístupní informace o tělesné teplotě pacientů a stavu teplotních senzorů.

Obsah

1. Úvod

1.1. Motivace

Základní motivací je zavedení moderních technologií do zdravotnictví, jejichž výsledkem je zvýšení komfortu jak pro pacienty, tak pro zdravotnický personál. Tak dojde ke zvýšení a zrychlení informovanosti personálu o stavu pacienta, a tím ke zkvalitnění zdravotní péče.

Nové technologie – výzkumné centrum Západočeské univerzity v Plzni v současnosti vyvíjí systém bezdrátových senzorů, které měří a zaznamenávají hodnoty tělesné teploty. Cílem vývoje těchto senzorů a systému ukládání a vizualizace dat je zcela nahradit nutnost používání klasických teploměrů pro měření tělesné teploty a ručního zaznamenávání hodnot do papírových archů.

Z testování systému, uskutečněného v Klatovské nemocnici, a.s., vyplynulo, že optimálním způsobem, jak zpřístupnit teploty pacientů personálu, je pomocí dotykového panelu, který bude umístěn v místnosti vrchní sestry, případně na dalších vhodných místech. K vizualizaci naměřených dat byla již dříve vyvinuta webová aplikace, která sloužila primárně pro detailní rozbor naměřených dat. Její použití ale v praxi není vhodné z důvodu odlišných požadavků na funkčnost aplikace.

Je vhodné vyvinout systém, který by fungoval na zařízeních s dotykovým panelem pro umístění na vhodná místa na nemocničním oddělení. Panely by měly personálu včas a přehledně poskytnout informaci o teplotách sledovaných pacientů a vhodně upozornit na teplotní výkyv a umožnit rychleji reagovat na aktuální zdravotní stav pacienta.

1.2. Cíle práce

Cílem této práce je vyvinout aplikaci pro dotykový panel s operačním systémem Android, která přehledně zpřístupní informace o tělesné teplotě pacientů a stavu teplotních senzorů. V rámci práce bude analyzován aktuální způsob sběru teplotních dat a jejich zaznamenávání a zpřístupnění pomocí existující webové aplikace. Bude popsáno, jak měření teplotními senzory probíhá a jaké jsou předpoklady ke správnému měření. Z existující analýzy akceptovatelnosti senzorů pacienty a hodnocení první fáze testování systému zdravotnickým personálem bude určena požadovaná funkčnost na aplikaci pro dotykový panel. Tato aplikace bude následně vyvinuta a otestována.

2. Fyziologická data, způsoby sběru a vizualizace

Fyziologie člověka je věda zabývající se fungováním lidského organismu. Mezi základní fyziologické funkce, které se u pacientů sledují, patří tělesná teplota, puls, krevní tlak a dýchání [1].

V této práci se budeme zabývat zejména měřením a sledováním tělesné teploty.

2.1. Tělesná teplota

Tělesná teplota je přirozená teplota organismu, která zaručuje správné fungování tělesných orgánů a reakcí, které v nich probíhají. Průměrná tělesná teplota u člověka se pohybuje mezi 35,8 °C – 37,3 °C. Tělesnou teplotu ovlivňuje věk, denní doba, tělesná aktivita, hormonální produkce, teplota a vlhkost okolí, příjem potravy a psychický stav. Zdravý organismus zachovává rovnováhu mezi produkcí a výdejem tepla.

Nejnižší teplotu má člověk brzy ráno (mezi 2. a 5. hodinou), nejvyšší odpoledne (okolo 18. hodiny). V průběhu dne kolísá, zvyšuje se při fyzické námaze nebo po požití potravy (teplota kolísá až o 2 °C [2]). Nejvíce tepla vytváří játra, ledviny a kosterní svalstvo, jejichž teplota může stoupnout až k 40 °C. Nejchladnější jsou periferní části těla (kůže a končetiny), které přizpůsobují teplotu okolí, a tím zabraňují vysokým ztrátám tepla. Největší vliv má teplota okolí na děti. Nejvíce náchylní k podchlazení jsou novorozenci kvůli relativně velkému povrchu těla a malému množství podkožního tuku. Starší lidé mají tělesnou teplotu často nižší než normu kvůli nižší tělesné aktivitě a kardiovaskulární kapacitě (termolabilita) [3].

Rozsah tělesné teploty

- pod 32 °C – smrt
- 32 – 36 °C – podchlazení (hypotermie)
- 36 – 37 °C – normální tělesná teplota (normotermie)
- 37 – 38 °C – zvýšená tělesná teplota (subfebrilie)
- 38 – 40 °C – horečka (febris)
- 40 – 42 °C – vysoká horečka (hyperpyrexie)
- nad 42 °C – smrt

Odolnost vůči nízkým teplotám je vyšší než vůči vysokým. Člověk snese bez následků i krátkodobý pokles teploty až na 21 °C (cca 20 min). Zvýšená tělesná teplota je přirozenou reakcí organismu na působení virů, bakterií a cizorodých látek – brání jejich rozmnožování a aktivizuje imunitní systém. Je-li teplota vnějšími vlivy zvýšena nad 40,5 °C (například při úpalu) a trvá-li delší dobu, selhávají termoregulační centra (v podhrbolí – hypotalamu, které je částí mezimozku).

2.2. Místa pro měření tělesné teploty

Hodnota naměřené teploty vždy záleží na místě na těle, kde se teplota měří. Obecně se rozlišuje:

- Teplota jádra, která je měřena vložením teploměru do tělesné dutiny, která poskytuje teplotu sliznice.
- Teplota povrchu, která je měřena na povrchu kůže a je smíšenou teplotou mezi teplotou jádra a okolní teplotou.

Rozdělení dle oblasti na lidském těle:

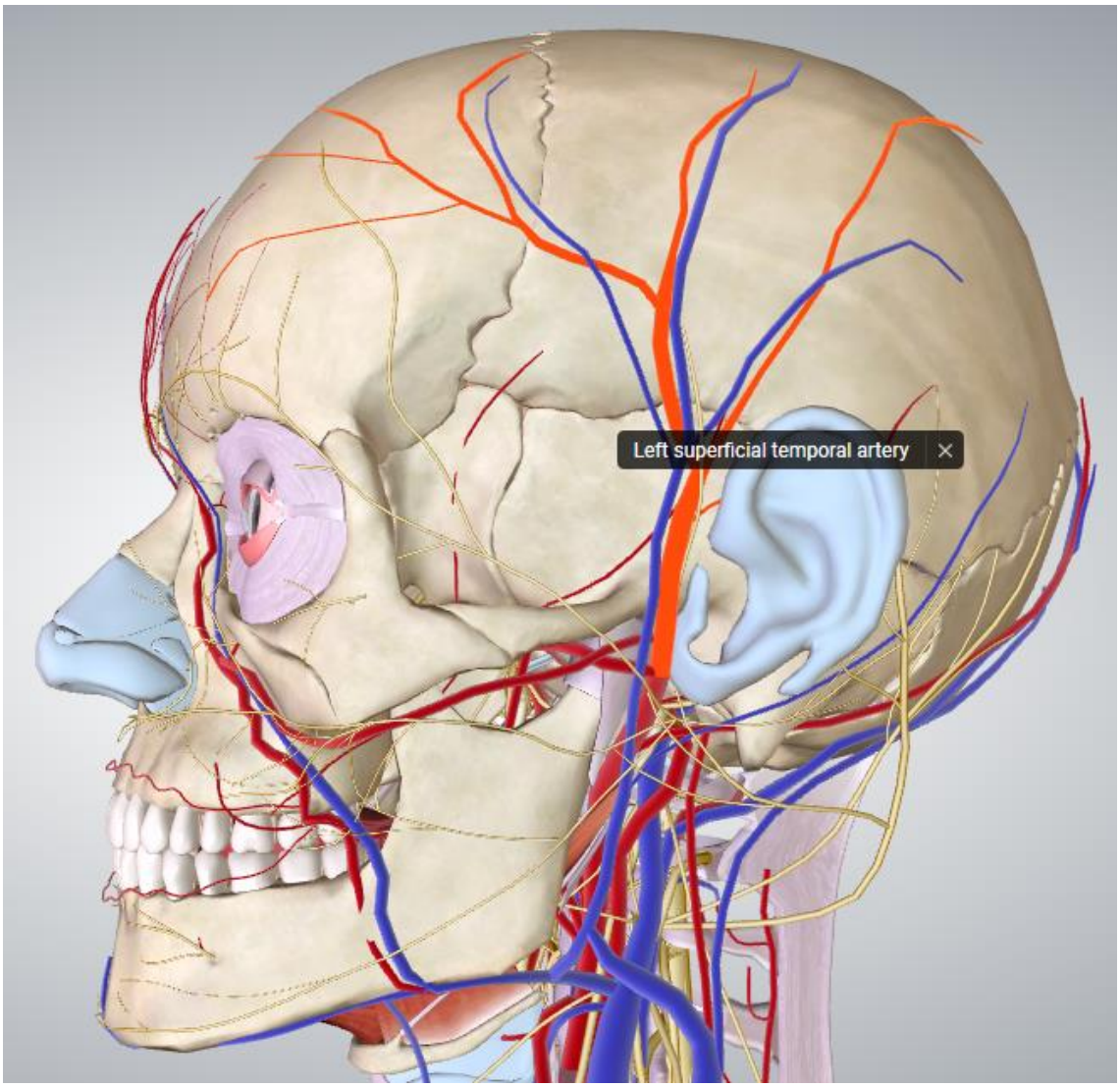
- Podpaží (axilla) – vhodné pro větší děti a dospělé. Měří se po dobu 10 min.
- Ústa (orální) – teplota o 0,1 – 0,3 °C vyšší než v podpaží. Průměrná tělesná teplota v klidu je v ústech 36,6 –37 °C. Nevhodné pro malé děti. Při měření v ústech pacient nesmí jíst a pít horké a studené 30 min před měřením. Teploměr se umisťuje pod jazyk a měří se 3 – 4 min.
- Konečník (rektální) – teplota o 0,5 °C vyšší než v podpaží. Zavádí se do konečníku do hloubky 2 – 4 cm a měří se 2 – 5 min.
- Vagína (bazální) – používá se ke zjištění doby ovulace. Teplota se měří ráno, než žena vstane. Teploměr se zavádí v poloze na zádech s pokrčenými dolními končetinami a měří se 8 – 10 min.
- Vnější zvukovod – teplota o 0,6 °C vyšší než v podpaží. Výhodou je rychlost měření, které trvá 2 – 3 sekundy.

- Třísla – měří se po dobu 10 min. Naměřená hodnota je stejná jako v podpaží.
- Měření na čele a ve spánkové oblasti.
- Intraabdominální – měření v jícnu či močovém měchýři.

Tělesná teplota se nejčastěji měří teploměrem v podpaží nebo v ústech. Měřená osoba by měla být ve fyzickém klidu ideálně 30 min před měřením, bez přehřátí po fyzické námaze. Měření by se také nemělo provádět bezprostředně po probuzení. Nejspolehlivější metoda je nicméně rektální měření, kde je normální teplotní pásmo mezi 36,2 – 37,7 °C [4].

Senzory, které jsou použity v rámci našeho systému, se umisťují do oblasti spánkového svalu. Touto oblastí prochází povrchová spánková tepna a její větve (arteria temporalis superficialis, obr. 2.1), které přivádějí krev ze srdce a mají tedy teplotu blízkou teplotě jádra. Tato oblast je používána pro měření teploty pomocí infračervených teploměrů, v rámci studií pro klinická hodnocení infračervených teploměrů byla prokázána použitelnost tohoto místa pro měření tělesné teploty [5] [6].

Senzor je speciálně navržen pro měření v této oblasti. Používá metodu kompenzace teploty okolí pro dosažení definované maximální odchylky od teploty jádra. Cílem těchto senzorů je poskytnout měření tělesné teploty srovnatelné přesnosti a spolehlivosti s používanými postupy měření v praxi.



Obrázek 2.1: Povrchová spánková tepna [7]

2.3. Typy teploměrů

Metody měření tělesné teploty ovlivňují stav pacienta, druh onemocnění a věk. Tomu se přizpůsobuje i volba teploměru tak, aby bylo měření co nejbezpečnější, nejpřesnější a zároveň neobtěžovalo pacienta.

- Lékařský teploměr maximální – bezrtuťový, měří teplotu v podpaží, třísle, v pochvě. Lékařským teploměrem měříme 5 – 9 min. Před měřením je potřeba zkontrolovat, zda je teploměr sklepán pod 35 °C. Základem bývá galium. Bezrtuťový teploměr maximální nahradil rtuťový teploměr, který se dle směrnice EU ve zdravotnických zařízeních již od roku 2009 používat nesmí, a to z důvodu vysoké toxicity rtuti.
- Digitální teploměr – plastový teploměr se zvukovým signálem. Výsledné měření trvá do 10 sekund.
- Bezkontaktní teploměr – pro měření teploty ve vnějším zvukovodu ucha a v oblasti čela. Měření je rychlé, většinou do 1 sekundy. Pracuje na bázi infračervené záření a je vybaven laserovým ukazovátkem.
- Páskový teploměr – funguje na principu zabarvení. Přikládá nebo lepí se na čelo. Tato metoda není považována za příliš spolehlivou [8], jde pouze o orientační měření tělesné teploty.
- Teplotní čidlo – ukazuje aktuální tělesnou teplotu na monitorovém systému. Je určené pro dlouhodobé měření teploty. Zpravidla se umisťuje na povrch kůže nebo sliznice. Využívá se na specializovaných jednotkách intenzivní péče (JIP) a anesteziologicko-resuscitačních odděleních (ARO).
- Elektronické teploměry (kontaktní) – šidítko, kožní teploměr na močovém katétru [9].
- Rychloběžný teploměr – používá se u dětí k měření teploty v konečníku.
- Rektální teploměr maximální – bezrtuťový nebo digitální.
- Invazivní metody – snímač zaveden do organismu. Cévka v tepně, sonda v močovém katétru.

2.4. Zaznamenávání tělesné teploty

Ve zdravotnických zařízeních se v praxi používají dvě metody měření tělesné teploty, a to měření digitálním teploměrem či bezkontaktním (infračerveným) teploměrem. V nemocnicích na standardních odděleních se sleduje teplota dvakrát denně, na odděleních ARO a JIP častěji. Dále se měří při výskytu příznaků, které může výkyv teploty způsobit. U pacientů se sklony k hypotermii (podchlazení organismu) či hypertermie (přehřátí organismu) se teplota sleduje častěji, například po dvou hodinách. Při podání léčiv či fyzickém chlazení se interval měření snižuje až na 30 minut. U termolabilních pacientů je potřeba měřit teplotu pomocí speciálních přístrojů kontinuálně [10].

V nemocnicích se tělesná teplota měří mezi 5. a 6. hodinou ranní z důvodu nejnižších hodnot v průběhu dne. „Pacienty brzké buzení obtěžuje. Někteří pacienti nemohou v noci spát a ráno jsou probuzeni jen kvůli měření teploty.“ [10] Personál se tedy snaží posunout dobu měření na pozdější hodinu a spojit jej s podáváním léků, hygienickou péčí či úpravou lůžek.

Naměřené hodnoty se zaznamenávají do dokumentace nemocného (dekurz, teplotní tabulka). Hodnota tělesné teploty se do dekurzu zapisuje ve formě číslice, např. T – 36,5 °C nebo do teplotní tabulky ve formě grafického záznamu teplotní křivky. Grafický záznam vznikne spojením jednotlivých naměřených hodnot, které se zaznamenají do teplotní tabulky v bodech a pak se spojí. Teplotní křivka umožňuje jednoduchý přehled o naměřených hodnotách v průběhu několika dnů nebo i týdnů. Dle zvyklostí ošetřovací jednotky je možné k záznamu použít červenou barvu. Tělesnou teplotu měřenou v konečníku můžeme zaznamenat také např. R – 37,2 °C.

2.5. Alternativy monitorování tělesné teploty

V dnešní době jsou vyvíjeny alternativní systémy pro monitorování tělesné teploty. Vhodným příkladem je 3M™ SpotOn™ [11], který primárně slouží k monitorování přesné vnitřní teploty v průběhu perioperačního procesu (zdravotní péče v období před, během a po operaci).

Tato metoda je neinvazivní a nahrazuje klasické invazivní metody jako pulmonální, jícnový, močový nebo rektální katétr. Ačkoliv většina invazivních přístrojů měří vnitřní teplotu přesně, jejich použití je omezené na operační sály. Naopak neinvazivní přístroje vnitřní teplotu nejsou schopny měřit, tudíž jsou naměřené hodnoty spíše odhadem.

Při použití se na čelo pacienta umístí jednorázový snímač (viz obr. 2.2), který má na sobě v průběhu celého perioperačního procesu. Snímač je připojen do monitorovacího přístroje kabelem, který lze odpojit a v případě potřeby opět připojit. Teplota se zobrazuje kontinuálně a výrobce uvádí v porovnání s pulmonálním katétrem odchylku nižší než 0,23 °C [12].

Obrázek 2.2: Ukázka použití systému 3M™ SpotOn™ [11]

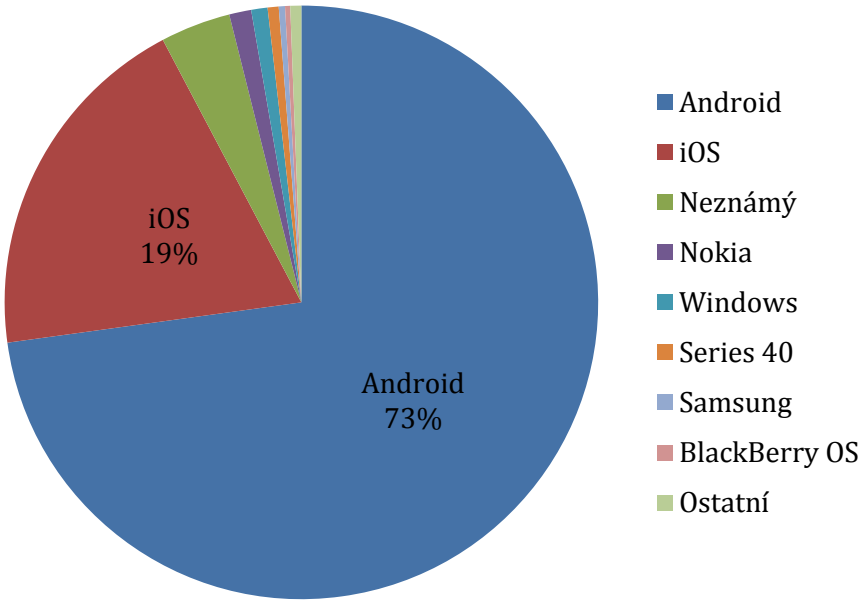
3. Chytré telefony a jiná chytrá zařízení

V této kapitole si stručně definujeme základní pojmy tematiky chytrých zařízení, jejichž základním rysem je běh na mobilním operačním systému, a charakterizujeme jejich vlastnosti, které pak ovlivňují vývoj aplikací. Dále si popíšeme jejich nejvíce rozšířené operační systémy pro mobilní zařízení.

3.1. Chytrá zařízení

Mobilní aplikace jsou provozovány na tzv. chytrých zařízeních (smart devices), za něž považujeme elektronické zařízení charakterizované hardwarovým a softwarovým vybavením, možností připojení k dalším zařízením nebo k síti prostřednictvím bezdrátových protokolů a interaktivním a autonomním chováním. Zahrnujeme mezi ně zejména chytré telefony (smartphones), tablety, hodinky (smartwatches), televize (smartTV) a další. Chytrá zařízení využívají operační systém a aplikační rozhraní, které umožňuje instalaci nebo úpravy programů [13].

V současnosti existuje jen několik mobilních operačních systémů, z nichž jen několik je masově rozšířeno. Obr. 3.1 znázorňuje poměry zastoupení používaných mobilních operačních systémů na chytrých telefonech v 1. čtvrtletí roku 2016.



Obrázek 3.1: Podíl mobilních operačních systémů, 06/2017 [14]

3.2. Mobilní platformy

3.2.1. Android

Operační systém Android, vyvíjený společnosti Google Inc. a Open Headset Alliance[1], je založený na jádře Linuxu. Jádro je navrženo pro běh na různém hardwaru a použití je nezávislé na chipsetu, velikosti a rozlišení obrazovky. Android můžeme vyjma chytrých telefonů nalézt i na tabletech, televizích, noteboocích, hodinkách či v automobilových systémech. V současnosti zastupuje přibližně 68 % podíl na trhu s chytrými telefony a téměř 34 % na trhu s tablety [15].

Android je dostupný jako open source software. Operační systém je vyjma koncových uživatelů nabízen též výrobcům zařízení a mobilním operátorům, kteří systém upravují pro vlastní účely. K tvorbě aplikací poskytuje nástroje pro jejich vývoj – Android Software Development Kit.

Často diskutovaným tématem je bezpečnost tohoto systému. Bezpečnostní rizika vyplývají z velkého množství výrobců zařízení, kdy je důsledkem zpožděné vydání bezpečnostním aktualizací (případně žádné bezpečnostní aktualizace pro zařízení vydaná před více než dvěma lety), využívání kódů třetích stran, které mohou obsahovat skryté funkce, nedostatečné obezřetného chování uživatele a podobně.

Stahování aplikací pro systém Android je dostupné zejména přes Google Play, který na počátku roku 2016 obsahoval okolo dvou milionů aplikací [16]. K publikaci aplikací skrze Google Play je potřeba mít placený účet pro vývojáře.

3.2.2. iOS

Druhým nejpoužívanějším mobilním operačním systémem je iOS společnosti Apple Inc. iOS je zároveň nejvíce rozšířeným operačním systémem na tabletech, kde jeho zastoupení činí 65%. iOS je proprietární systém založen na UNIXu a je zjednodušenou mobilní verzí Mac OS X, který je známý z osobních počítačů Apple.

[1] Open Headset Alliance je seskupení výrobců mobilních telefonů, telekomunikačních operátorů a technologických firem (Google, HTC, Sony, Dell, Intel, T-mobile, aj.).

iOS byl původně určen jen pro mobilní telefony iPhone, později začal být dodáván i na tablety iPad, multimediální přehrávač iPod, televizorech AppleTV a další zařízení, která jsou vyráběna výhradně firmou Apple. Oproti většině dalších operačních systémů jej nelze licencovat na zařízeních jiných výrobců. I přes tuto skutečnost se iOS těší velké oblibě.

Apple pro vývoj nativních aplikací poskytuje iPhone Software Development Kit. Primárními jazyky pro vytváření iOS aplikací jsou Objective C nebo Swift, přičemž některé prvky mohou být psány i v C či C++. Nejznámějším vývojovým prostředím je Xcode. Jiné jazyky mohou být také používány, ale z důvodu omezení od Apple nejsou obecně k dispozici v App Store, což je služba pro stahování aplikací, obdobně jako Google Play pro Android. V polovině roku 2015 bylo v App Store dostupných 1,5 milionů aplikací [17].

3.2.3. Windows Phone

Windows Phone je proprietární mobilní operační systém vyvíjený společností Microsoft. Po systémech Android a iOS je třetím nejvíce používaným operačním systémem na chytrých telefonech. Primárně jej nalezneme na zařízeních výrobců Microsoft Mobile/Nokia, též HTC a Samsung. Zaměřuje se na chytré telefony a tablety s úhlopříčkou do 8 palců. Od roku 2015 vystupuje pod značkou Windows 10 Mobile a přináší sjednocení a integraci s počítačovým operačním systémem Windows 10. Zařízení s čekávaným systémem Windows 10 Mobile nicméně přichází na trh až nyní.

Vývoj aplikací odráží koncept technologií společnosti Microsoft. Pro počítačový operační systém je typické uživatelské prostředí s konceptem dlaždic (Metro), který je v aktuální verzi mobilního operačního systému nazýván Modern UI. Oficiálním jazykem pro vývoj aplikací je jazyk C#, patřící do rodiny .NET jazyků. Programování v C# vyžaduje znalosti objektově orientovaného programování stejně jako u Javy pro Android aplikaci a Objective C či Swift u iOS. Programovat se dají i tzv. univerzální aplikace společně pro mobilní Windows Phone a klasická Windows 8, kde kód sdílí asi 95 % vlastností. To je výhodné, neboť lze stejnou aplikaci zároveň vyvíjet pro počítačová i mobilní Windows. To platí pro vývoj na bázi C#, C++ i HTML/JS [18].

Stahování aplikací probíhá přes Microsoft Store, který v polovině loňského roku zahrnoval přibližně 340 tisíc aplikací [19].

3.3. Charakteristika chytrých zařízení

V dalším textu se budeme věnovat výlučně chytrým telefonům a tabletům. Zatímco základními prvky ovládání počítačů jsou klávesnice a myš, chytré telefony jsou ovládány dotykem. Mají ale i další vlastnosti, kterým je potřeba vývoj aplikací přizpůsobit tak, aby bylo použití snadné, uživatelsky příjemné a plynulé.

3.3.1. Obrazovka

Telefony mají v současnosti velikost úhlopříčky displeje přibližně od 3" do 6" (palců), přičemž nejběžněji používané jsou okolo 5". Běžné tablety mají úhlopříčku 7" až 10". Skupina chytrých telefonů o fyzické velikosti od 5" do 7" je někdy také označována jako „phablet“. Dalším parametrem je pak rozlišení displeje. Při vytváření aplikace musíme dbát na to, aby zobrazené informace a ovládání bylo přizpůsobeno různým velikostem zařízení a byla tak zaručena pohodlnost práce s aplikací.

Dále je potřeba vzít v úvahu, aby aplikace vypadala vizuálně příjemně. Zatímco na klasických počítačích je monitor téměř vždy orientovaný „na šířku“, mobilní telefon či tablet je možno otočit na výšku (tzv. portrait) i na šířku (tzv. landscape). Mobilní telefony mívají výchozí orientaci na výšku, u tabletů je to různé. Je proto vhodné, aby vyvíjená aplikace byla schopna reagovat při změně orientace zařízení a zachovat konzistentnost uživatelského rozhraní – ať už úpravou velikostí či přeskládáním komponent.

3.3.2. Hardware

Při návrhu aplikace musíme zajistit, aby byl běh telefonu stabilní, zároveň aby byl stále schopen spouštět další aplikace a běžela zcela plynule. Je nutné se vyvarovat prodlev, které by ovlivnily ostatní funkce zařízení, zejména schopnost přijímat příchozí hovory. Aplikace tedy musí správně uvolňovat alokovanou paměť.

V současné době je běžné mít k instalaci aplikací interní úložný prostor v řádu několika GB, nicméně využití paměti v řádu stovek MB pro aplikaci se dá pochopit například u navigací, kdy má uživatel v paměti uložené mapy několika států. Rozhodně platí, že čím menší aplikace, tím je pro uživatele pohodlnější.

Mobilní zařízení jsou napájeny bateriemi, jejichž kapacita je značně omezena. Aplikace by tedy měla být co nejúspornější vzhledem. Na vysokou náročnost na baterii aspirují zejména aplikace, kde je potřeba udržovat síťové připojení nebo se provádějí výpočty a operace. Ve spojitosti s výdrží baterie je potřeba zmínit další požadavek na aplikace – v případě vypnutí zařízení nebo při násilném ukončení aplikace, se musí aplikace ukončit v konzistentním stavu tak, aby operace byly nějakým způsobem dokončeny a při opětovném spuštění aplikace nedošlo k selhání. Těchto náhlých situací je samozřejmě vícero, například výpadek internetového připojení, ztráta GPS signálů.

3.3.3. Další vybavení

Chytré telefony jsou samozřejmě vybaveny přístupem do sítí Wi-Fi a 3G, což přináší obrovské možnosti využití internetu. Dnes je docela běžné, že majitelé chytrých telefonů a tabletů mají mobilní internet, ovšem většinou s limitovaným objemem stahovaných dat. Android aplikace by tedy měla stahovat co nejmenší množství dat. Mezi další vybavení patří například kamera, mikrofon, akcelerometry, GPS[2] apod.

3.4. Využití ve zdravotnických zařízeních

Využití chytrých telefonů se rozšířilo do všech možných vrstev, což je dáno též množstvím a rozmanitostí aplikací, které lze používat. Pojem „chytré zařízení“ se ve spojitosti se zdravotnictvím vyskytuje čím dál častěji. I v této specifické oblasti, kde se původně používaly speciální nástroje, dnes vznikají aplikace, které lze ve spojení s jiným zařízením využívat jako vyšetřovací nástroj. Různými zařízeními je například možné měřit krevní tlak, kontrolovat úbytek a nárůst tělesného tuku či

[2] GPS neboli Global Positioning Systém je globální družicový polohový systém. Umožňuje určit geografickou polohu na Zemi s přesností na jednotky metrů.

svalové hmoty. [20] Svůj potenciál mají chytrá zařízení též při vyšetřování pacientů v terénu nebo v částech světa s nedostatečnou lékařskou péčí.

Mezi takovéto nástroje patří například AliveCor[3] – ruční EKG, kde jsou elektrické impulsy měřeny ze srdce a zachyceny snímacími destičkami, které měřená osoba pevně drží oběma rukama. Na chytrém zařízení aplikace poté zobrazí srdeční frekvenci a rytmus. Tak lze včas rozpoznat problém se srdeční činností, což je důležité zejména u rizikových pacientů.

Mezi další nástroj patří ruční ultrazvuk MobiSnate[4], která se fyzicky připojuje ke smartphone a pracuje se s ní jako s klasickým ultrazvukem. [20]

V neposlední řadě zmíníme Biomeme[5], což je malá vyšetřovací laboratoř obsahující testovací kazetu, která je schopna ve vzorcích krve a moči na základě chemických reakcí rozeznat původce nemoci. Do zařízení se připojí chytrý telefon, který vyhodnocená data vizualizuje (viz obr. 3.2).

Obrázek 3.2: Biomeme two3 [21]

V současnosti zmíněné vyšetřovací nástroje nedosahují přesností jako nemocniční přístroje, nicméně jejich potenciál lze spatřovat v jednodušším ovládání, možnosti provádění rutinních vyšetření bez potřeby návštěvy lékaře, efektivnějších možnostech ukládání a sdílení dat či mobilitě vyšetřovacích přístrojů.

[3] https://www.alivecor.com/

[4] http://www.mobisante.com/

[5] http://biomeme.com/

4. Současný systém

Pro vizualizaci nasnímaných teplotních dat z bezdrátových senzorů byla vytvořena webová aplikace Biomon Portál. Tato aplikace byla vyvíjena pro použití z webového prohlížeče na PC. Aplikace slouží pro monitorování stavu senzorů a detailnímu procházení naměřených dat a jejich komentování. Původně byla aplikace vyvíjena pro použití jak při vývoji systému, tak pro zdravotnický personál (sestrám a lékařům). Pozdější nasazení v praxi ukázalo, že přístup k datům přes PC je nutné doplnit aplikací pro přístup z dotykového panelu, který bude trvale a přehledně zobrazovat aktuální stav pacientů.

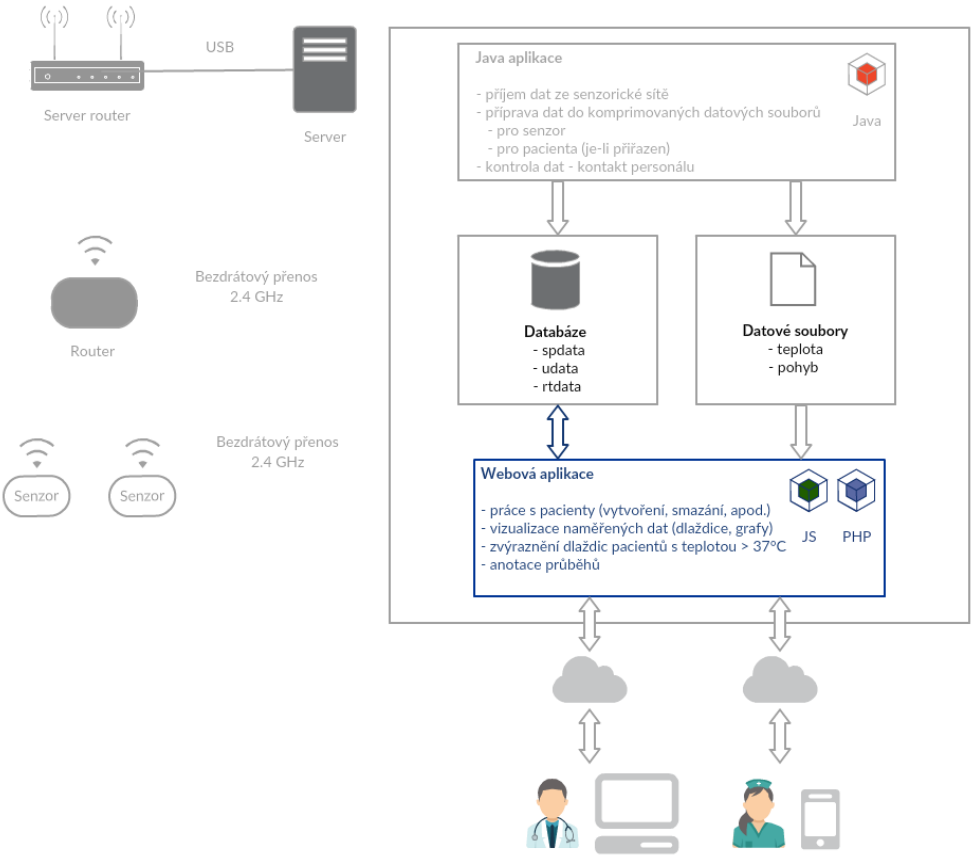
V následující kapitole se budeme věnovat analýze současné webové aplikace, jejíž vývoj započal v rámci bakalářské práce Jiří Brát [22] a která byla autorkou této práce rozšiřována v rámci projektu KIV/OPSWI. Na základě analýzy navrhneme úpravy a vybereme funkce, které jsou vhodné k implementaci v aplikaci pro dotykový panel.

4.1. Měřicí systém

Prostřednictvím webového prohlížeče uživatel vyvolává dotazy nad databází a načítá datové soubory. Databáze obsahuje zejména identifikační data, zatímco datové soubory obsahují konkrétní fyziologická data. Získaná data jsou zpracována a přehledně zobrazena v uživatelském prostředí. V otevřeném webovém prohlížeči je nastavena průběžná automatická aktualizace dat a zobrazených komponent.

Datovou vrstvu tvoří jednak souborová databáze a jednak komprimované datové soubory. Komunikaci serveru a webového prohlížeče zajišťuje program v jazyce PHP. Webové uživatelské prostředí je psáno v jazycích HTML, CSS a JavaScript.

Na obr. 4.1 lze vidět zařazení webové aplikace do architektury celého měřicího systému.



Obrázek 4.1: Schéma měřicího systému

4.2. Zdroj dat

4.2.1. Senzory

V současné době jsou vyvíjeny senzory pro snímání, přenos a vyhodnocení fyziologických signálů, a to zejména pro použití ve zdravotnickém a pečovatelském prostředí. Umožňují měřit teplotu a míru pohybu. Senzor má velikost cca 3 cm x 1,5 cm a je oválného tvaru s rovnou měřicí plochou, která se používá ke kontaktnímu snímání teploty v místě měření.

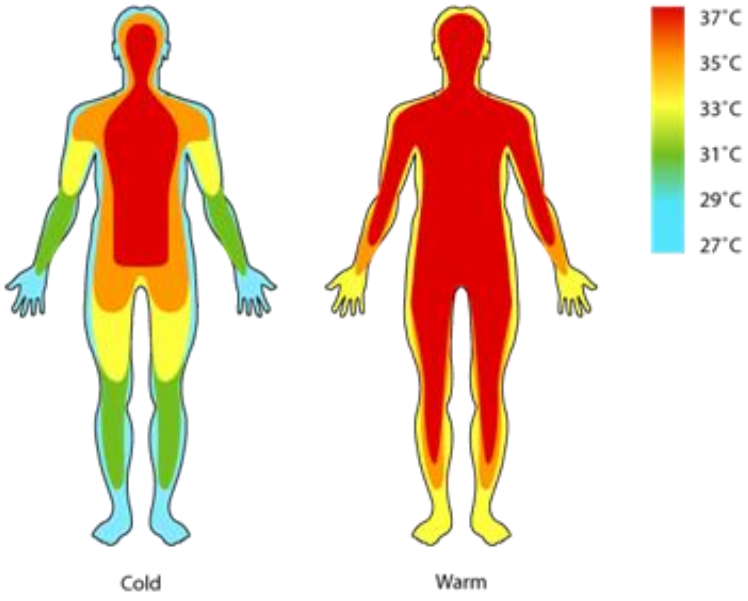
Pacientovi je na kůži v oblasti spánkového svalu aplikován senzor. Testy ukázaly, že kvalitní připevnění je problematické zejména u výrazně se potících pacientů. Dostatečných výsledků je dosahováno při fixaci pomocí oboustranné pásky od společnosti 3M s certifikací pro trvalý styk s kůží. Při aplikaci senzoru na pacienta při běžné pokojové teplotě dojde ke stabilizaci měřené teploty během několika minut. Tato doba a odchylka měřené teploty od teploty jádra závisí

na algoritmech kompenzace vlivu okolí, které jsou v současné době stále optimalizovány. Pro možnost klinického použití je nutné vypracovat studii srovnávající teplotu měřenou senzorem s referenčním klinickým měřením v dostatečné počtu případů.

Jak jsme již zmínili v kapitole 2, hodnota naměřené teploty vždy záleží na místě na těle, kde se teplota měří. Obecně se rozlišuje:

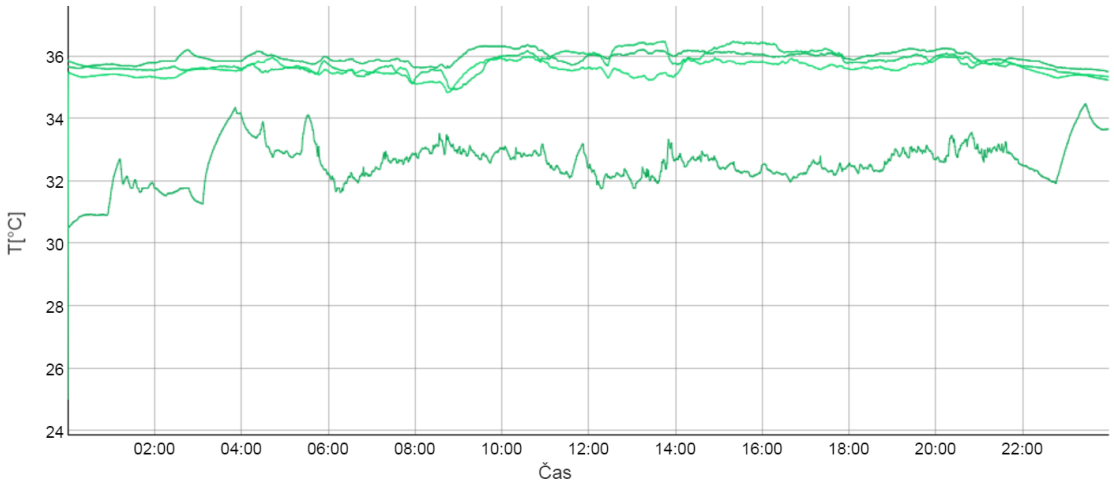
- teplota jádra, která je měřena vložením teploměru do tělesné dutiny, která poskytuje teplotu sliznice,
- teplota povrchu, která je měřená na povrchu kůže a je smíšenou teplotou mezi teplotou jádra a okolní teplotou.

Hodnota naměřené teploty je tedy závislá na umístění senzoru. Nejvíce se teplotě jádra přibližuje hlava, krk a okolí srdce, v těchto místech je tedy vhodné sledovat tělesnou teplotu (viz obr. 4.2). Vhodná místa pro umístění senzorů pro měření tělesné teploty jsou spánky, krk a podpažní jamka. Zde se teplota pohybuje okolo 36 °C (pro zdravého člověka při normální aktivitě). Klinicky použitelných míst pro měření teploty lidského těla je pouze několik. Pro sledování tělesné teploty pomocí použitého sensoru se jako nejlepší jeví umístění na spánkovém svalu. Je ale možné připevnit senzor například v blízkosti operační rány, změna teploty v tomto místě pak může indikovat kupříkladu zánět rány.



Obrázek 4.2: Hodnoty tělesné teploty při chladné a teplé okolní teplotě [23]

Použitím senzorů v oblasti kolem operační rány se zabýval projekt TAČR TD020094 Využití senzoriky pro zefektivnění péče o stárnoucí a hendikepovanou populaci. Na obr. 4.3 můžeme vidět průběh teplot ze tří senzorů umístěných v okolí rány (3 průběhy s nejvyšší teplotou). Čtvrtý teplotní průběh na obrázku je ze senzoru umístěného v oblasti podpažní jamky. U tohoto senzoru je vidět, že měření je zatíženo značnou chybou způsobenou v největší míře nekvalitní fixací senzoru na kůži a dále změnami v zakrytí senzoru při pohybu rukou.



Obrázek 4.3: Vizualizace originálních naměřených dat (zdroj: Biomon Portál)

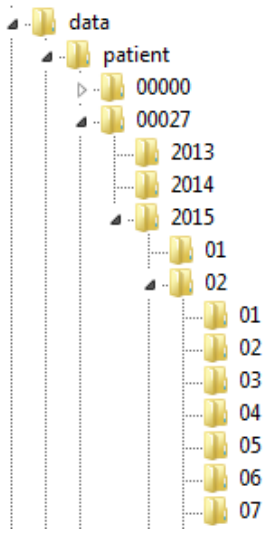
4.2.2. Datové soubory

Senzorická teplotní data jsou uložena v binárních souborech. Tyto soubory vytváří aplikace, která přijímá senzorická data. Potřebné soubory si webová aplikace stáhne vždy při zobrazení stránky. Každému senzoru je přiřazen jeden binární soubor na jeden den.

Adresářová struktura (obr. 4.4) pro uložení datových souborů vypadá následovně: /data/patient/pid/yyyy/mm/dd_sid_ddmmyyyy.dat, kde pid je id pacienta, sid je id senzoru, dd je den, mm je měsíc, yyyy je rok.

Struktura binárního souboru:

Každý teplotní záznam se skládá z posloupnosti 32 bitů. Prvních 17 bitů (neznaménkově) udává čas v sekundách od počátku dne. Následujících 15 bitů (znaménkově, první bit udává znaménko) udává teplotu v setinách stupně Celsia.



Obrázek 4.4: Adresářová struktura datových souborů

4.2.3. Databáze

Jako databázový systém je používán SQLite 3. Data o pacientech, senzorech a uživatelích jsou uložena ve třech databázových souborech: spdata, rtdata a udata, v závorkách jsou uvedené atributy.

spdata – obsahuje tabulky sensors (sid, pid, desc), patients (pid, name, create_time, delete_time, deleted), annotations

rtdata – obsahuje realtime data o senzorech (id, aktuální teplota, napětí baterie v senzoru, apod.).

udata – obsahuje tabulku users (id, username, password, role, deleted, create_time, delete_time).

4.3. Webová aplikace

4.3.1. Funkcionalita

Přihlašovací stránka a registrace

Úvodní stránka, po úspěšném přihlášení pomocí jednoduchého formuláře je uživateli umožněn vstup do aplikace. Účty jsou zakládány prostřednictvím registrace. V případě zapomenutého přihlašovacího jména nebo hesla jej lze obnovit prostřednictvím zadané e-mailové adresy.

Hlavní stránka

Hlavní stránka má dvě komponenty.

V horní části obsahu stránky se nachází tabulka, která poskytuje přehled všech aktivních senzorů v systému. Její sloupce obsahují jméno pacienta, teplotu senzoru, identifikátor senzoru a obslužná tlačítka daného pacienta. Tabulka mění pořadí řádků podle nejvyšších teplot senzorů. V druhé části stránky je umístěn velký graf obsahující teplotní křivky všech senzorů, které aktuálně měří. Každý senzor je zobrazen jinou barvou, respektive senzory jednoho konkrétního pacienta mají stejnou barvu, jen jiný odstín. Graf ukazuje data od půlnoci aktuálního dne.

Historie měření

Zobrazuje seznam všech pacientů, kteří byli do systému zavedeni, a jejich měření probíhá, nebo již bylo ukončeno smazáním pacienta ze systému. Umožňuje přechod na detail měření konkrétního pacienta.

Detail měření

Prohlížení historie měření pacientů je možné jak pro aktuální měření, tak pro již ukončené měření. Detail obsahuje graf se senzory pacienta, do nějž lze vypisovat anotace pro identifikaci událostí souvisejících s měřením v daném čase.

Založení nového pacienta

Přidání pacienta do systému umožňuje asociaci pacienta s teplotními senzory. Zakládat nové pacienty je povoleno uživateli s rolí “Admin”.

Uživatelé

Na stránce uživatelů je možné zavádět do systému nové uživatele a stávajícím měnit heslo v případě zapomenutého hesla. Seznam uživatelů a funkce změny hesla jsou dostupné pouze uživatelům s rolí “Admin”.

Informace o senzorech

Stránka senzory obsahuje tabulku senzorů zavedených v databázi a informací o nich. Tato stránka je přístupná pro uživatele s rolí “Admin”.

4.3.2. Třídy uživatelů

V aplikaci existují 3 třídy uživatelů: „Uživatel“, „Sestra“ a „Admin“.

Uživatel má přístup k naměřeným datům, může zakládat a mazat pacienty.

Sestra sleduje teploty pacientů na hlavní stránce a v detailu pacienta. Zavádí do systému nové pacienty s odpovídajícími senzory a ukončuje jejich měření.

Admin má přístupné všechny funkce systému. Navíc spravuje seznam uživatelů, kteří mají do systému přístup. Může potvrzovat/zamítat registrace uživatelských účtů, zakládat/mazat uživatele a stávajícím měnit uživatelská jména či role.

4.3.3. Použité technologie

Jedná se o webovou aplikaci klient – server. Serverová část připravuje data z existující části senzorického systému. Klientská část vytváří uživatelské prostředí a vizualizuje data připravená serverem.

Klient

Struktura kódu aplikace psána pomocí frameworku *Backbone.js*. Komunikace a výměna dat mezi klientem a serverem probíhá ve formátu JSON.

Design aplikace je založen na sadě nástrojů *Bootstrap* 2. Tabulky jsou reprezentovány komponentou *DataTables.js*. Knihovna použitá pro vizualizace dat v grafu je *dygraphs*.

Server

Pro serverovou část je použit framework PHP SlimFramework, který má architekturu REST[6]. Pro přístup ke zdrojům dat (databáze) aplikace je použita metoda přístupu (get, post, delete, put) a identifikátor URI.

4.4. Zhodnocení webové aplikace

Architektura systému je navržena tak, že není potřeba ji měnit i při implementovaných změnách a rozšířeních.

Ačkoliv je kód aplikace založen na frameworku Bootstrap, jehož předností je možnost responzivního vývoje, nebyl design implementován pro malé obrazovky. Problémem jsou zejména mobilní zařízení s malými obrazovkami s vysokým rozlišením (telefony) a velkými obrazovkami s malým rozlišením. Neexistuje žádná detekce typu zařízení a obsah stránky přetéká mimo viditelnou oblast.

Současná aplikace je webová aplikace vytvořená a optimalizovaná pro použití ve webovém prohlížeči. Toto řešení není vhodné pro použití na dotykovém panelu ve zdravotnickém a pečovatelském prostředí.

Hlavním požadavkem na aplikaci je minimalizace činností spojených s obsluhou systému. Studie v Klatovské nemocnici, a.s. (viz kap. 4.5) zahrnující nemocniční personál a pacienty ukázala, že hlavní potřebou sester je přehledná a stále viditelná informace o aktuální tělesné teplotě pacientů a stavu alarmů a senzorů, kterou je možné realizovat pomocí dotykového panelu umístěného v místnosti sester, nebo na dalších vhodných místech (například sesterna, chodba, v blízkosti dveří do pokoje pacientů). Informace nemusí být nutně vztažena ke jménu pacienta, ale spíše se jako vhodnější jeví identifikace pomocí čísla pokoje a lůžka. Tento přístup přináší několik výhod a to minimalizaci činností při vytváření pacientů v systému a přirozená anonymizace dat.

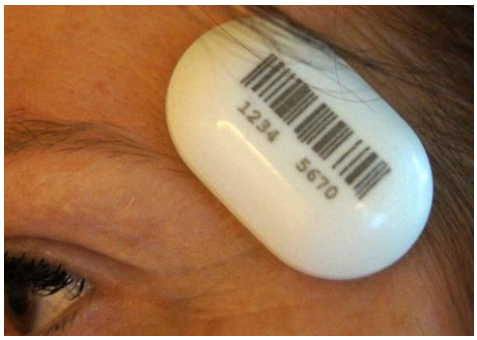
[6] Více na https://www.zdrojak.cz/clanky/rest-architektura-pro-webove-api/.

4.5. Zhodnocení senzorického systému

V rámci testování tohoto senzorického systému na měření byla publikována studie „Measuring Body Temperature Wirelessly: Patients' and Personnel's Acceptance in a Hospital Environment" [24], která zkoumala přijetí zařízení pacienty a zdravotnického personálu v nemocničním prostředí, konkrétně v Klatovské nemocnici, a.s.

Výzkum trval 2 měsíce (září až říjen 2016) a měření se zúčastnilo 27 pacientů ve věku 21 až 72 let (průměrný věk 50 let), z nichž bylo 10 mužů, 15 žen a dva neuvedli pohlaví. Každé měření trvalo 3 dny, což je i průměrná doba hospitalizace.

Senzor byl umístěn na spánek a k pokožce připevňován pomocí oboustranné lepicí pásky (viz obr. 4.5). Oblast hlavy je vhodná zejména z důvodu vysoké teploty, která se hodnotou nejvíce z celého povrchu těla blíží teplotě jádra (viz kap. 4.2.1). Spánek též poskytuje místo, kde se nachází tenká spánková kost/spánkový sval, kde lze senzor bezproblémově uchytit a kde není výsledek ovlivňován tělesnými proporcemi pacienta (například podkožním tukem).



Obrázek 4.5: Připevněný senzor [24]

Po ukončení měření byli pacienti dotazováni v následujících oblastech. Pro všechny odpovědi viz přílohu A.

- Obecné hodnocení bezdrátového teploměru
- Hodnocení testování bezdrátového teploměru
- Hodnocení dlouhodobého nošení teploměru

Rozhodná většina účastníků (25) hodnotí měření tělesné teploty jako přínosné. V otevřené odpovědi ohledně pozitiv a negativ pak byla zmíněna

možnost okamžité informace o zdravotním stavu (4) a oceněno vyhnutí se nutnosti časného probuzení pacienta (2). Většina účastníků (22) obecně hodnotí zkušenost s nošením teploměru jako pozitivní, přičemž jeden účastník vyjádřil negativní pocity. Méně účastníků (17) má pocit, že teploměr může být zaveden do nemocničního prostředí v jeho aktuální formě, někteří (8) jsou o aktuální formě skeptičtí. Hlavním zjištěným problémem, který pravděpodobně způsobil negativní vnímání vyššího počtu respondentů na možnost realizace teploměru, bylo odlepování senzoru v průběhu nošení.

Jedenáct respondentů nošení teploměru vůbec nezaznamenávalo, 12 jej zaznamenalo příležitostně, 4 příležitostně nebo často. Problémy se spánkem při nošení nezaznamenal nikdo z dotazovaných.

Ohledně designu teploměrů (velikost a tvar) většina (16) účastníků vyjádřila spokojenost, někteří (7) design neokomentovali a 3 jej označili za nevhodný. První limitací byla podle respondentů výška teploměru, druhou celková velikost a poslední tvar. Dle zdravotnického personálu mohla být velikost překážkou ne v komfortu nošení, ale například při převlékání nebo česání.

Jako hlavní problém celého výzkumu se ukázalo odlepování senzoru při dlouhodobém nošení. U jednoho pacienta se objevilo podráždění kůže v důsledku použití lepicí pásky.

Zdravotní personál (2) vyjádřil sympatie k ideji bezdrátového měření tělesné teploty. V jednom případě byly kritizovány moderní metody měření teploty, které neposkytují přesné údaje. Navíc na oddělení, kde výzkum probíhal, se měří teplota pouze dvakrát denně, a proto nelze sledovat vývoj teplot, které mohou poukazovat například na zánět. V druhém případě byla oceněna snaha nechávat pacienty déle spát a nebudit je kvůli měření teploty. Ani jedna ze sester nezmínila stížnosti pacientů nebo pocit studu pacienta při nošení teploměru. Na základě odpovědí se v průběhu testování nevyskytly žádné vážné překážky. Pacienti nebyli omezováni v jakékoliv aktivitě (například chůze, možnost opustit prostor oddělení). Bezdrátový teploměr též nebyl problémem při lékařských zákrocích. Obě sestry potvrdily, že teploměr by mohl být nasazen do lékařského prostředí v jeho aktuální formě. [24]

4.6. Možné případy užití

Z testování plyne několik možností, jak by se informační systém k záznamu a vizualizaci naměřených hodnot mohl v praxi využívat. Navrhneme dvě z nich s charakteristickými body.

První vychází z potřeb nemocničního oddělení Klatovské nemocnice, a.s., jejichž prioritou je anonymizace pacientových dat, minimalizace času stráveného při ovládání a zadávání údajů do systému, které jsou již zadány do nemocničního informačního systému.

1. Každé lůžko má vlastní senzor

- Identifikace pacienta bude v systému zcela anonymní, namísto jména nebo rodného čísla se zadá pokoj a lůžko (například pokoj 2A, lůžko 3)
- Data o pacientovi se do systému vůbec nezadávají
- Aplikace je propojena na nemocniční informační systém, kde jsou již data o pacientovi zadána (osobní údaje, pokoj, lůžko, datum hospitalizace)
- Naměřené hodnoty tělesné teploty se automaticky přiřazují do karty pacienta (přiřazení a smazání senzoru se děje v nemocničním informačním systému, existuje samostatné datové propojení systému senzorů na nemocniční informační systém, které v této práci není popisováno)
- Měření každého senzoru běží nepřetržitě, měření se nezakládají a nemažou

Druhá možnost je vhodná v případě, že není žádoucí napojení na nemocniční informační systém, je vhodné flexibilně přiřazovat senzory jednotlivým pacientům nebo lůžkům (například jsou-li lůžka často neobsazená a je tak zbytečné, aby senzory zaznamenávaly data nepřetržitě).

2. Senzory se přiřazují podle potřeby

- Identifikace pacienta v aplikaci
- Jednotlivá měření se zakládají a ukončují včetně zadání informací k senzoru specifických pro dané měření
- Aplikace není napojena na nadřazený informační systém

5. Aplikace pro Android

V této části si stručně popíšeme základní principy tvorby aplikací pro platformu Android. Shrneme si možné přístupy k vývoji aplikací, zmíníme, kdy je dané řešení vhodné či nikoliv. Prozkoumáme nejpoužívanější vývojová prostředí, jejich odlišnosti a programovací jazyky pro vývoj aplikací. Dále se podíváme na strukturu zdrojových kódů pro Android aplikace, a co musíme zajistit pro publikaci a distribuci vyvinuté aplikace.

5.1. Přístupy k implementaci aplikací

V této kapitole se zaměříme na popis a srovnání různých přístupů k vývoji aplikací pro mobilní platformy.

5.1.1. Mobilní web

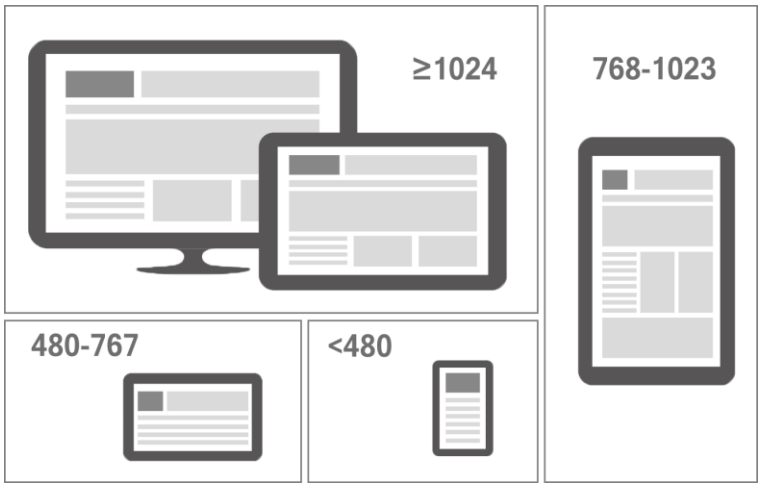
Mobilní web je univerzální formou, která funguje napříč všemi platformami. Jedná se o podobu webových stránek, která je přizpůsobena velikosti a rozlišení displeje chytrých telefonů. Konkrétně pro webové stránky pak existují dva způsoby řešení – jednak mít dva nezávislé weby pro desktopové počítače a pro telefony, jednak jednotný responzivní web s detekcí zařízení s drobnými úpravami. Samozřejmostí přitom je, že při práci s webem musí být zařízení po celou dobu připojeno k internetu.

Responzivní[7] webdesign jsou webové stránky s HTML stylováním takovým, že se daná stránka zobrazí na zařízeních různé velikosti a uživatel se tak dostane k jakémukoliv obsahu (obr. 5.1). Server webu nerozlišuje mezi jednotlivými typy zařízení a všem posílá stejná data. Stránku upravuje pouze prohlížeč podle velikosti okna, ve kterém se web zobrazuje. Rozpoznání vlastností zařízení zajišťují zejména Media Queries ve specifikaci CSS3. Následující CSS kód zajišťuje změnu barvy pozadí pro zařízení s rozlišením obrazovky do 500 pixelů.

[7] Přizpůsobující se prostředí (např. rozlišení, orientaci displeje, nastavení velikosti písma).

```
@media only screen and (max-width: 500px) {
      body {
            background-color: lightblue;
    }
}
```

První náznaky responzivity se objevily na webových stránkách Audi.com v roce 2001, která se přizpůsobovala různým šířkám prohlížeče [25]. Ještě několik let zpět byl plně responzivní web zastoupen jen několika vybranými stránkami. Dnes se již tvorba responzivního webu poměrně bezproblémová, jelikož existují knihovny, které úpravu usnadňují, a zároveň jsou odstraňovány rozdíly mezi jednotlivými internetovými prohlížeči.



Obrázek 5.1: Responzivní web [26]

Výhodou při vývoji mobilního webu je rychlost vývoje, kdy je web upravován jen změnou CSS kódu. Tento kód je pak univerzální pro všechny platformy.

Mobilní web poskytuje jednotnou prezentaci obsahu uživatelům s různými druhy zařízení. Má ovšem poměrně negativní stránku – možnosti a kvalita interakce s uživatelem jsou do jisté míry omezené. Prvním nedostatkem je, že při prohlížení stránky vzhledem k architektuře webových stránek Request & Response[8] vzniká časová prodleva mezi odesláním požadavku a přijetím dat, respektive jejich zobrazením ve vykreslovaném obsahu. Toto zpoždění lze samozřejmě snížit dostatečnou rychlostí internetového připojení, ne však zcela.

[8] Základní metoda vzájemné komunikace mezi vícero počítači, kdy jeden počítač vyšle požadavek na určitá data a druhý počítač odpoví na požadavek.

Z důvodu dotykového ovládání chytrých zařízení může být problém s grafickým ovládacím prvkem mouseover[9], jelikož takový stav neexistuje. Také vyplňování formulářů na dotykové klávesnici může být nepohodlné a je vhodné nutné vepisování textu eliminovat na minimum. V případě malé velikosti komponent se může stávat „překliknutí“ uživatele, kdy se nedotkne požadované komponenty. Je proto potřeba ověřit, že na všech velikostech zařízení jsou tlačítka a další ovládají prvky dostatečně velké tak, aby nevznikal například tzv. „fat-finger problem“[10].

Jelikož k mobilnímu webu se nepřistupuje jako do klasické mobilní aplikace, nýbrž přes webový prohlížeč, nemůže spolupracovat s vestavěnými funkcemi zařízení (například kamera či upozornění na obrazovce telefonu) či s ostatními aplikacemi tak, jako nativní aplikace. Řešením tohoto nedostatku se však zabývají vývojáři společnost Google, kde se v budoucí verzi mobilního operačního systému Android M bude internetový prohlížeč chovat více jako mobilní aplikace. To by zajišťovala technologie zvaná „instalovatelné webové aplikace“, kdy se aplikace nainstaluje do zařízení tak, že uživatel nepozná, že se jedná o internetový prohlížeč. Rozdíly mezi aplikací a mobilním webem nebudou tedy tak markantní [27].

Mobilní web není potřeba, ale zároveň není ani možnost distribuovat přes obchody třetích stran – u Androidu Google Play.

5.1.2. Hybridní aplikace

Hybridní aplikace mohou být zajímavým kompromisem. Z vizuálního hlediska jsou mezistupněm mezi klasickou nativní aplikací a mobilním webem. Mají přístup k hardwarovým schopnostem chytrého telefonu a zároveň je lze využívat na více platformách. Hybridní aplikace mají většinou serverovou část, kde je realizována většina funkcí, a nativní část optimalizovanou pro dané zařízení dostupnou na příslušném aplikačním storu [28]. Uživatel, který do svého zařízení instaluje nativní část, může využívat různé serverové části (tedy fakticky pracovat s různými

[9] Událost přejetí kurzorem myši po elementu.

[10] Fat-finger problem je definován jako potíž při prohlížení webové stránky na dotykové obrazovce, kdy tlačítka na stránce jsou menší než prst, kterým se tlačítko ovládá.

aplikacemi). Z hlediska návrhu můžeme rozlišovat dva druhy: WebView a kompilované aplikace.

WebView aplikace využívají interního prohlížeče k zobrazení obsahu webu, jež je postaven na technologiích HTML, CSS a JavaScript. JavaScript na straně klienta pak funguje jako prostředník s API platformy. WebView je pak „obalen" pomocí kódu nativní aplikace, který umožňuje otevřít okno, vestavěné do aplikace, v němž se otevře web identifikovaný pomocí URL nebo zobrazí jiný HTML kód. Ve srovnání s mobilním webem je aplikace uložena přímo v zařízení. V aplikaci je pak přístupný totožný obsah jako při přístupu z internetového prohlížeče. Vzhled stránky tak zůstane zachován a operačnímu systému se přizpůsobí logika aplikace. Dle zaměření aplikace může být snížena potřeba datových přenosů při komunikaci se serverem, přičemž se snižuje odezva na akce uživatele.

Kompilované Android aplikace jsou takové, jejichž kód je vytvořen v jiném jazyce než Java, ale při kompilaci jsou do nativního jazyku převedeny. Výhodou při vývoji hybridních aplikací je využití existujícího frameworku pro tvorbu rozhraní mezi aplikačním kódem a API rozhraní. To umožňuje zrychlení vývoje aplikace v případě již existujícího kódu pro webové stránky nebo v případě, že chceme vyvíjet aplikaci pro více platforem. Hybridní kompilovaná aplikace má totožný kód pro všechny platformy. Ten se při kompilaci převede do nativního kódu pro Android.

Za zmínku stojí framework Ionic SDK, který je určený pro vývoj hybridních aplikací. Mnozí by mohli namítnout, že aplikace je pro všechny platformy vizuálně stejná, a nebude tak ladit s konvencemi designu různých platforem (například Windows Phone se značně liší od Android a iOS). Android používá jiné přechody mezi stránkami, používá jiné prvky uživatelského rozhraní, nadpisy, dialogy a tak dále. Ionic automaticky přizpůsobuje vzhled komponent podle specifických guidelines[11] platformy.

Nejrozšířenějším řešením vývoje hybridních aplikací jsou aplikace založené na jazycích HTML, CSS, JavaScript, přičemž kód je postaven na frameworku

[11] Sada doporučených postupů.

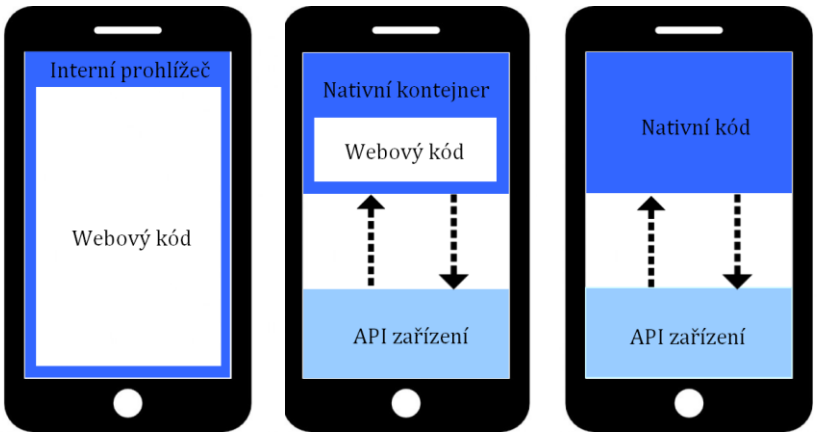
Cordova/PhoneGap. Ten je tvořen v nativním kódu platformy a obsahuje WebView přes celou obrazovku aplikace. Zároveň lze komunikovat s API dané platformy.

Za nejsilnější stránkou hybridního vývoje stojí zejména multiplatformnost, díky které lze aplikace vyvíjet 2 až 3x levněji [29]. Testovat aplikaci lze jednoduše v prohlížeči, v případě propojení s nativním kódem v emulátoru dané platformy.

Negativní stránkou hybridních aplikací mohou být zvýšené požadavky na hardware. Tento problém se však v současnosti postupně minimalizuje se zlepšujícím se výkonem telefonů a prohlížečů.

5.1.3. Nativní aplikace

Nativní aplikace jsou programované pro konkrétní operační systém. Jsou rychlé a mohou využívat hardwarové schopnosti chytrého telefonu (kamera, GPS, kalendář, apod.). Tyto aplikace mohou fungovat off-line a šetřit tak výdrž baterie, což je v současnosti stále problém chytrých telefonů.



Obrázek 5.2: Srovnání mobilního webu, hybridní aplikace a nativní aplikace

Nativní aplikace mohou často poskytovat lepší výkon. Aplikace jsou psány v nižších programovacích jazycích než hybridní aplikace nebo mobilní web a nejsou tak závislé na internetovém připojení a komunikaci se serverem.

Chceme-li vyvíjet jednu aplikaci v nativním jazyce na více platforem, zvyšují se nároky na programátory, kteří musí umět vícero syntakticky odlišných jazyků: například při vývoji aplikace na Android Javu a na iOS Objective C či Swift. Když je poté zapotřebí aplikaci opravovat a aktualizovat, opravy se provádějí v různých

kódech na různé platformy. Vzhledem k tomu, že se vyvíjí tatáž aplikace vícekrát, je výsledná cena aplikace vyšší.

Obr. 5.2 zobrazuje srovnání zmíněných přístupů k vývoji aplikace.

5.1.4. Technologické parametry ovlivňující výběr přístupu

Cílová platforma a náklady na vývoj

Nativní aplikace jsou vyvíjené na míru pro každé zařízení a operační systém. Je to cesta náročnější, nákladnější a s obtížnějším prováděním změn, než u hybridních aplikací a mobilního webu. Vývoj a testování je potřeba řešit zvlášť pro různé operační systémy, různé verze systémů, velikosti obrazovek, a funkcionality zařízení. Naopak může vzniknout uživatelsky nejpříjemnější a nevyspělejší aplikace. Dalším aspektem mohou být též požadavky na design aplikace. Čím více chceme vlastní design, který by navíc byl na různých platformách stejný, tím více se vyplatí mobilní web a hybridní aplikace.

Cílová skupina uživatelů

Zvolená architektura aplikace závisí též na cílové skupině uživatelů. Budeme-li vytvářet řešení pro několik tisíc uživatelů, je potřeba orientovat se na nativní aplikace (například aplikace pro mobilní bankovnictví). Otázkou také je, kolik uživatelů z cílové skupiny si vůbec aplikaci nainstaluje a kolik jich bude aplikaci opravdu využívat. Je zde též možnost vytvořit mobilní web či hybridní aplikaci jako mezistupeň před implementací nativní aplikace. Na mobilní web lze snadno navázat Google Analytics, pomocí kterého se dají zjistit informace, zda by byl o nativní aplikaci zájem.

Existence funkčního kódu

V případě, že vytváříme mobilní aplikaci se stejnou funkcionalitou jako u webové aplikace, a máme již funkční a vyhovující webovou aplikaci, případně se některé komponenty dají znovu využít, je vhodné vybrat si stejné technologie. Lze tím ušetřit čas a zároveň snížit náklady na opakovaný vývoj obdobného kódu. Obdobně platí, například máme-li webovou aplikaci, jež má serverovou část, kterou můžeme využít pro běh požadované aplikace, pak je hybridní aplikace vyhovující alternativou.

Využití nativních funkcí zařízení

Požadujeme-li, aby vyvíjená aplikace spolupracovala s nativními funkcemi chytrého zařízení (například GPS signál či čtečku QR[12] kódů), pak je potřeba zvolit jiné řešení, nežli mobilní web, který k nativním funkcím nemá požadované oprávnění.

5.2. Programovací jazyky pro Android

5.2.1. Java

Java je oficiální jazykem pro vývoj aplikací pro Android. Velká část systému Android je též psaná v Javě a API je navrženo tak, aby mohlo být voláno především z Javy. Java programy lze nelézt na mnoha typech zařízení počínaje od chytrých telefonů, přes desktopové počítače, Raspberry Pi až po mainframové počítače[13]. Kód se nekompiluje do nativního kódu procesoru, ale do byte code[14] pro Java Virtual Machine (JVM). Úkolem těchto virtuálních strojů je interpretovat strojový kód a převést jej na speciální běhové prostředí. Výhodou je pak zejména nezávislost na platformě, kde se aplikace spouští. Virtuální stroj pro Android aplikací do verze 4.4 se nazývá Dalvik, který pracuje v režimu just-in-time a převádí tak byte code při každém spuštění aplikace. Z této skutečnosti plynou větší požadavky na výkon a na čas spuštění aplikace. Nástupcem od Android verze 5.0 je Android Runtime (ART), jehož běhové prostředí pracuje v režimu ahead-of-time, kdy se pro koncové zařízení přeloží kód již při instalaci. Cílem je rychlejší spouštění aplikace, aplikace má zároveň větší požadavky na úložný prostor a čas instalace.

Pro vytváření a testování Android aplikací je potřeba nainstalovat Android Software Development Kit (SDK), který obsahuje API knihovny a nástroje potřebné k sestavení aplikace. Nejčastěji používaná vývojová prostředí jsou Android Studio a Eclipse.

---

[12] QR kód neboli kód rychlé reakce je prostředek pro automatizovaný sběr dat.

[13] Počítač používaný pro kritické aplikace, často zahrnující zpracovávání velkých objemů dat.

[14] Byte code neboli bajtkód jsou instrukční sady navržené pro realizaci snadno přenositelných aplikací pro běh na konkrétní platformě.

5.2.2. Corona

Jednou z alternativ k Javě je Corona SDK, multiplatformní framework určený pro vývoj mobilních a desktopových aplikací, zejména 2D her. Kód programu se píše v jazyce Lua, procedurálním programovacím jazyce navrženém jako skriptovací jazyk. Corona obsahuje emulátor, který spustí kód bez nutnosti kompilace. Pro vytvoření spustitelného .apk souboru se spustí vzdálená online kompilace, která uloží aplikaci lokálně. Stažení Corona SDK je zdarma, pro různé funkce nebo možnost volání nativního Android API se kupuje licence na měsíční bázi.

5.2.3. Apache Cordova

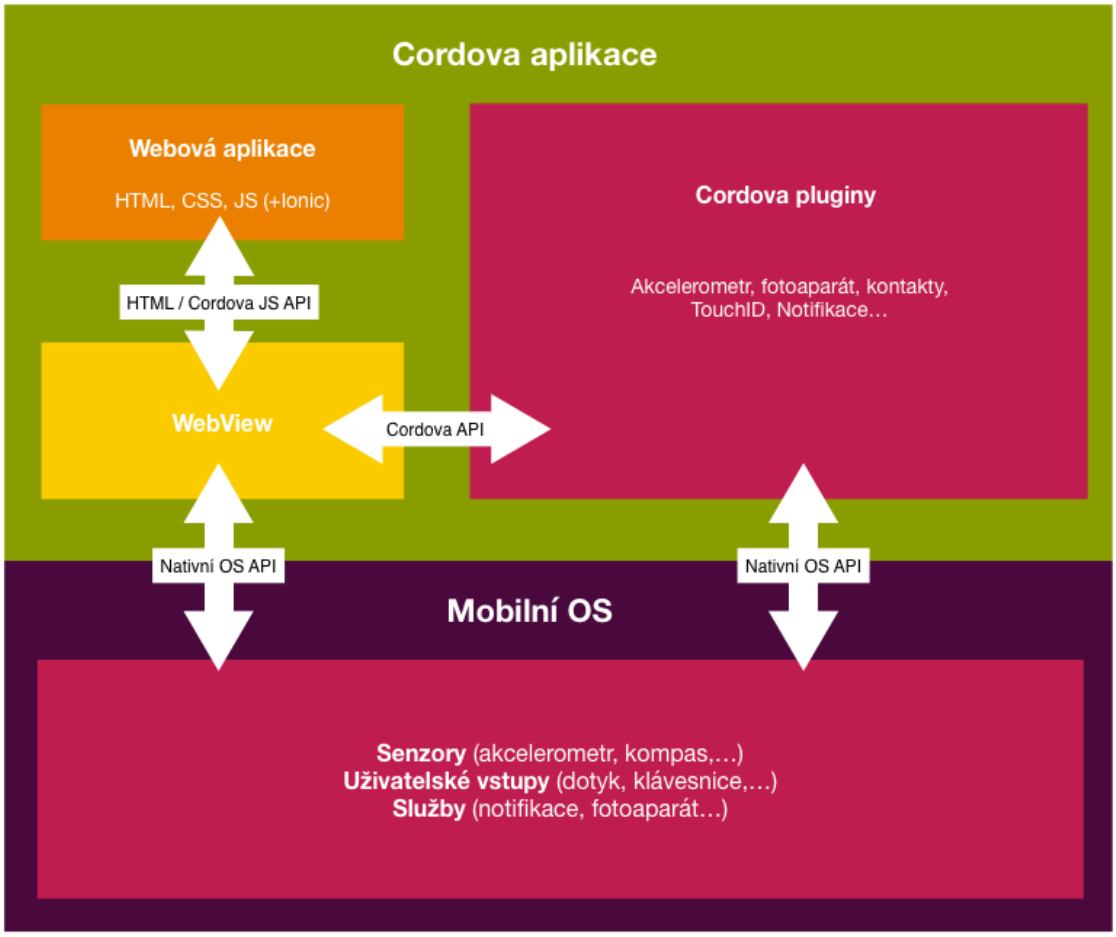
Další možností vytvoření Android aplikace jsou open-source frameworky Apache Cordova (obdobou je PhoneGap[15], který je na frameworku Cordova založen). Staví na standardních webových technologiích HTML, CSS a JavaScript, a tím umožňují snadný multiplatformní vývoj. Apache Corodova je k dispozici pro platformy Android, iOS, Windows Phone, Blackberry, Ubuntu, Firefox OS, Tizen, Amazon FireOS. Při komplikaci programu se aplikace „obalí“ kódem specifickým pro konkrétní mobilní platformu a prováže se s nativním API k umožnění přístupu k funkcím chytrých zařízení, jako je internet, akcelerometr, kamera, GPS, lokální úložiště, aj.

Použití Cordovy je vhodné v následujících případech:

- aplikace využívá WebView a zároveň je potřeba mít přístup k nativním funkcím chytrého telefonu
- z existující webové aplikace potřebujeme vytvořit mobilní aplikaci, která se navíc dá distribuovat v obchodech s aplikacemi (Google Play, App Store)
- cílem je vytvořit aplikaci, která se dá využívat na vícero mobilních platforem

[15] Adobe PhoneGap je distribucí Apache Cordova. V současnosti se funkcionalitou liší minimálně (například v ovládání přes příkazovou řádku, nebo v možnosti vzdáleného sestavení aplikaí), ale v budoucnu se počítá s odlišným vývojem vzhledem k samostatné distribuci PhoneGap.

Vizuální obsah aplikace je reprezentován WebView, které se dále ovládá pomocí JavaScriptu, případně hybridní aplikací – kombinací WebView a nativních komponent. Samotná aplikace je implementována jako webová stránka, která se spouští z WebView obaleného v nativní aplikaci. Cordova pluginy poskytují rozhraní pro komunikaci s nativními komponentami a s API zařízení, což umožňuje volat nativní kód přímo z JavaScriptu. Architekturu aplikace založené na Cordově znázorňuje obr. 5.3.



Obrázek 5.3: Architektura Apache Cordova [30]

Cordova oproti ostatním zmíněným přístupům vzhledem k architektuře neobsahuje nástroje pro navrhování UI aplikace, pro webové aplikace ale existuje nespočet takových knihoven a frameworků (například jQuery Mobile [31], AngularJS [32] nebo Ionic [33]).

5.2.4. Xamarin

Pomocí Xamarinu, který je od nedávna vlastněn společností Microsoft, lze vytvářet nativní aplikace pro mobilní operační systémy Android, iOS s Windows využitím kódové základny programovacího jazyka C#. Xamarin poskytuje přístup ke všem nativním rozhraním API, nativním uživatelským rozhraním jednotlivých platforem i specifickým rozhraním pro jednotlivé platformy. Výhodou toho nástroje je rychlý a efektivní vývoj mobilní aplikace pro více platforem, jejichž aplikační logika a grafické rozhraní jsou totožné. To snižuje nároky na znalosti specifických řešení jednotlivých platforem. Základní logika psaná v C# a grafické rozhraní, které se vytváří v nástroji Xamarin Forms pomocí jazyka XAML[16], je striktně odděleno. Výsledný kód v nativní formě dané platformy se vytvoří při kompilaci kódu bez nutnosti většího zásahu do kódu pro specifickou platformu.

Aplikace lze vyvíjet na platformách Windows ve vývojovém prostředí Visual Studio nebo OS X v prostředí Xamarin Studio. V poslední době přibývají další nástroje pro vývoj, jako například Xamarin Insights (pro monitoring), Xamarin Test Cloud (pro jednotkové a UI testy) či Xamarin Profiler (pro monitoring paměti). Licence Xamarinu je zdarma pro omezenou velikost zdrojového kódu aplikace a s omezenými možnostmi využití různých nástrojů. Placené licence začínají na 25 USD/měsíc.

5.3. Vývojové prostředí

K programování Android aplikací je k dispozici mnoho vývojových prostředí. Mezi nejrozšířenější patří Android Studio a Eclipse s ADT[17] pluginem, dále pak IntelliJ IDEA a NetBeans. První dvě zmíněná prostředí si popíšeme a porovnáme.

Android Studio bylo představeno v roce 2013 firmou Google jako oficiální vývojové prostředí výhradně pro platformu Android. Je založené na IntelliJ IDEA a je k dispozici zdarma pro všechny uživatele na platformách Windows, Mac OS a Linux. Oproti Eclipse má kvalitnější funkci doplňování kódu a menší nároky

[16] XAML neboli eXtensible Application Markup Language – značkovací jazyk pro tvorbu grafických rozhraní aplikací.

[17] Android Developer Tools

na výkon počítače. Součástí instalačního balíčku je kromě Android Studia IDE také Android SDK, kompilátor a sadu emulátorů. Android Studio má integrovaný sestavovací systém Gradle, jehož součástí je široké spektrum emulátorů chytrých zařízení Nexus.

Eclipse je open source vývojová platforma pro jazyk Java. Pro umožnění vývoje pro Android je potřeba doinstalovat pluginy ADK (Android SDK), ADT (Android Development Tools) a dále vybraný emulátor. Eclipse používá Apache Ant založený na XML či Maven. Uživatelské rozhraní je velmi robustní, a to zejména z důvodu univerzálnosti tohoto IDE. Nicméně Eclipse s ADT pluginem v roce 2015 ztratily podporu vývojářů Androidu a nedoporučuje se nadále používat [34].

Vývojová prostředí Android Studio a Eclipse ADT mají odlišná názvosloví a adresářovou strukturu projektu. Srovnání zobrazuje tab. 1.

Eclipse ADT	Android
Workspace	Project
Project	Module
Project-specific JRE	Module JDK
Classpath variable	Path variable
Project dependency	Module dependency
Library Module	Library
AndroidManifest.xml	app/src/main/AndroidManifest.xml
assets/	app/src/main/assets
res/	app/src/main/res/
src/	app/src/main/java/
tests/src/	app/src/androidTest/java/

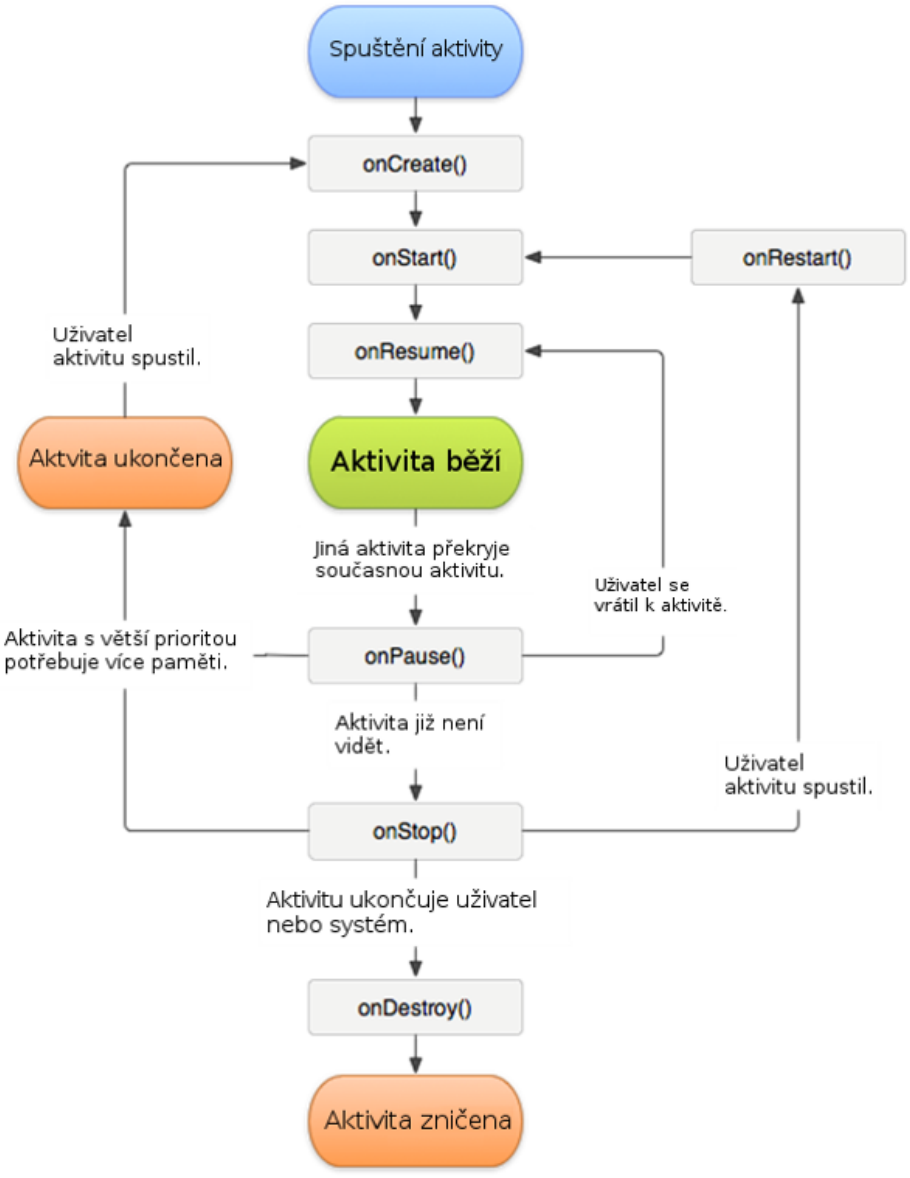
Tabulka 1: Adresářová struktura projektu Android Studio a Eclipse [35]

5.4. Komponenty aplikace

Aplikace pro platformu Android sestávají ze čtyř typů komponent:

Activity

Activity obsahuje grafické uživatelské rozhraní (GUI) pro interakci s uživatelem odpovídající jedné obrazovce, stránce, formuláři. Aplikace má zpravidla více aktivit, mezi kterými lze přepínat a předávat data. Aktivitou může být například úvodní stránka, přihlašovací formulář, stránka produktu. Samotný operační systém se pak stará o běh aktivit a v různých situacích volá různé metody. Životní cyklus aktivity zobrazuje obr. 5.4.



Obrázek 5.4: Životní cyklus aktivity [36]

Services

Service (služba) je proces běžící dlouhodobě na pozadí a umožňující provádět akce (stahování, připojení k serveru, výpočet). Nemá uživatelské rozhraní. Služba se může nacházet ve třech stavech:

- Component calls – inicializace služby pouhým zavoláním nebo navázáním komponenty na službu.
- Service is running – služba vykonává funkci na pozadí.
- Service is shut down – služba byla ukončena sama nebo komponentou, podle spuštění služby.

Content providers

Content provider je aplikační rozhraní, které poskytuje přístup k datům a je tedy cestou, jak sdílet data mezi aplikacemi (např. kontakty) i mezi jednotlivými aktivitami. Data mohou být umístěna v souborech, databázi, na webu, apod. Rozhraní content providers má metody podobné jako při práci s databází – insert, update, delete, query.

Broadcast Intent Receivers

Broadcast receiver je komponenta sloužící k naslouchání oznámení z vnějšku i zevnitř aplikace. Existují systémové (slabá baterie) a vlastní (upozornění z aktivity).

Intent je jakýmsi komunikačním mostem mezi activities, broadcasts receivers či services. Je to jednoduchý objekt, který může obsahovat primitivní data. Explicitní intent přímo určuje, kterou třídu chce spustit.

```
Intent i = new Intent(context, LoginActivity.class);
```

Implicitní intent definuje pouze záměr aplikace, ale způsob už nikoliv. Může se jednat například o otevření adresáře kontaktů či URL stránky ve webovém prohlížeči.

```
Intent i = new Intent(Intent.ACTION_PICK, Contacts.CONTENT_URI);
```

5.5. Manifest

Na operačním systému Android běží každá operace aplikace ve vlastním sandboxu, což je bezpečnostní mechanismus, který slouží pro oddělení běžících procesů. Pokud aplikace potřebuje přístup k systémovým zdrojům, datům jiné aplikace či soukromým datům uživatele, musí o to explicitně požádat prostřednictvím systémových oprávnění v souboru AndroidManifest.xml. Při instalaci aplikace se pak uživateli zobrazí seznam požadovaných oprávnění, bez jeho odsouhlasení nelze aplikaci nainstalovat.

Dále je v tomto souboru deklarována každá komponenta aplikace, na základě deklarací systém ví, co lze jak volat a jakým způsobem může komponenta spolupracovat s okolím.

V tomto souboru se dále nastavují parametry a informace aplikace – název, package aplikace, minimální požadované Android API, seznam používaných knihoven apod.

Soubor AndroidManifest.xml je uložen v kořenovém adresáři aplikace.

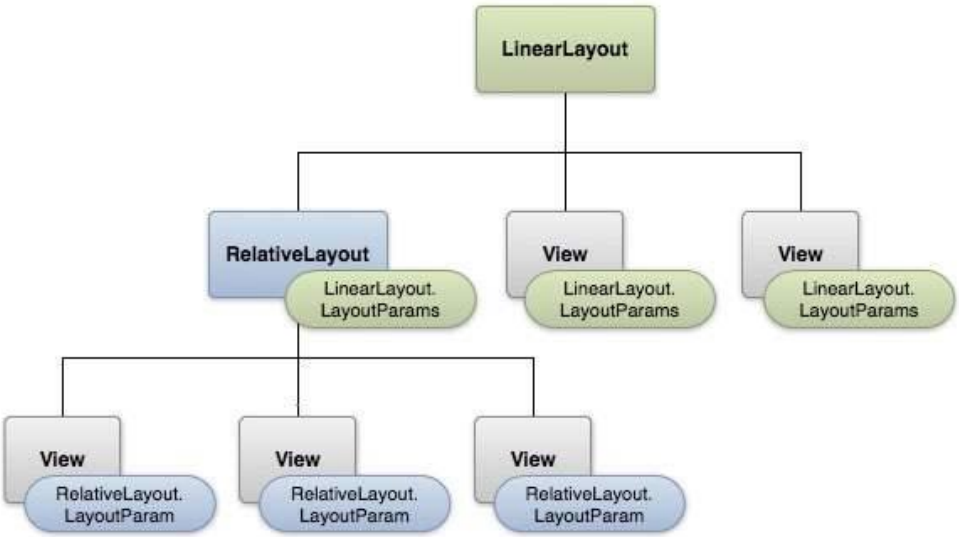
5.6. Uživatelské rozhraní

Z pohledu konstrukce výše uvedených komponent existují další, které definují jejich logiku a propojení.

Základním stavebním prvkem pro uživatelské rozhraní je View (neboli pohled) – objekt, který zaujímá obdélníkovou plochu obrazovky zařízení. Obsahuje ovládací prvky jako například tlačítka, textová pole a vytváří tedy interaktivitu mezi uživatelem a zařízením. Zodpovídá za vykreslování zpracování událostí.

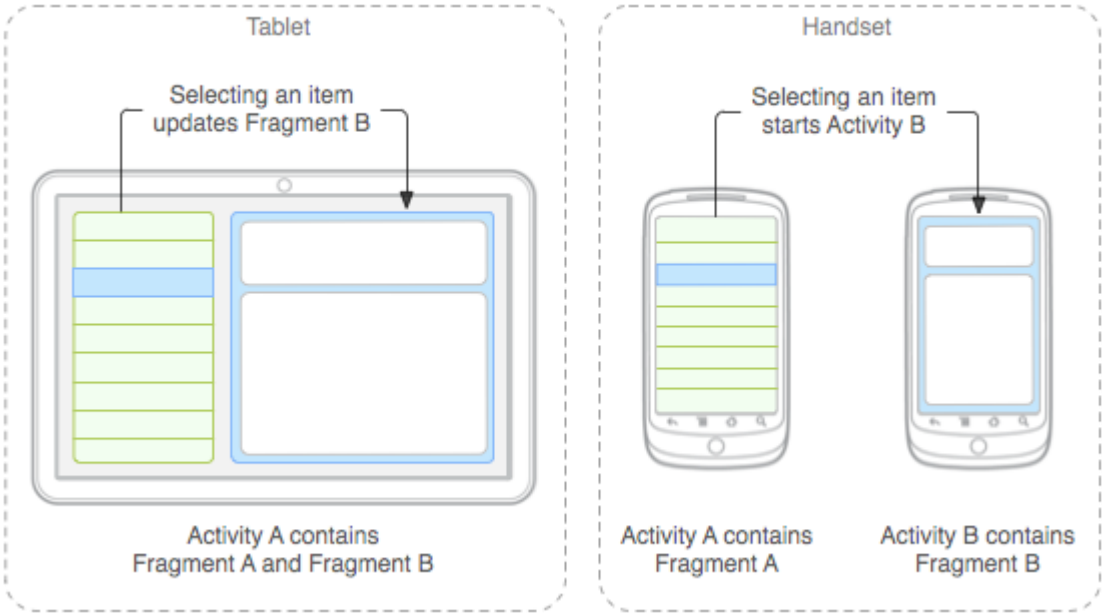
Views jsou seskupovány do ViewGroup, což je neviditelný container, který definuje jejich uspořádání (Layout). View je tedy potomek Layout (viz obr. 5.5).

Layout je definován v XML souboru typu, který definuje rozložení prvků rozhraní dané aktivity. Společně tedy tvoří vzhled obrazovky aplikace.



Obrázek 5.5: Struktura Views a Layouts [37]

Dále lze využívat prvky typu Fragment, který představuje ucelenou část aktivity s vlastní logikou i View. Z těchto vlastností Fragmentu jejich umožňuje různé rozložení prvků, díky čemuž se dá vytvořit vícepanelové rozložení obrazovky (viz obr. 5.6).



Obrázek 5.6: Vícepanelové využití prvku typu Fragment

Veškeré resources jsou uloženy v projektovém adresáři /res a jeho podsložky mohou obsahovat například XML soubory Layouts, utvářející společně s aktivitami vzhled obrazovky, definice menu, obrázky či lokální datové soubory.

5.7. Oprávnění aplikace

Operační systém Android je založen na sadě oprávnění, které aplikace potřebují ke svému běhu. Účelem je ochrana integrity systému a soukromí uživatele. Každá aplikace při instalaci nebo aktualizaci musí explicitně zažádat o povolení přístupu k dané operaci či funkci zařízení. Systém může automaticky oprávnění udělit, pokud nevyhodnotí riziko narušení soukromí uživatele, nebo se systém dotáže na udělení práv přímo uživatele.

Potřebná práva aplikace jsou deklarována v souboru AndroidManifest.xml, který se nachází v kořenovém adresáři každé aplikace. Například aplikace, která vyžaduje oprávnění k zasílání SMS, bude obsahovat v manifestu kód [38]:

```
<manifest xmlns:android="http://schemas.android.com/apk/res/android"
        package="com.example.snazzyapp">

    <uses-permission android:name="android.permission.SEND_SMS"/>

    <application ...>
        ...
    </application>

</manifest>
```

Od Android verze 6.0 uděluje uživatel práva při běhu aplikace místo při instalaci. Kromě zrychlení instalace aplikací má uživatel větší přehled nad tím, co aplikace vykonává. Ve starších verzích byl uživatel dotázán na sadu potřebných oprávnění a instalace v případě zamítnutí (byť by uživatel zamítnul jen některou operaci) nebyla provedena. V nové verzi může zamítnout přístup k jednotlivým operacím či funkcím.

5.8. Distribuce aplikace

Nativní aplikace pro Android lze distribuovat skrze Google Play (obdobně pak App Store pro Apple). Může se jednat i o komerční aplikaci, za kterou jsou uživatelé ochotni zaplatit. U mobilního webu nebo jiných webových variant je potřeba distribuci a propagaci vyřešit jiným způsobem. Druh distribuce ale vždy záleží na účelu aplikace a cílové skupině uživatelů.

6. Návrh aplikace pro dotykový panel

V této kapitole na základě průzkumu současné webové aplikace a analýze vyhodnocení akceptovatelnosti systému v praktickém použití sestavíme specifikaci požadavků. Předmětem specifikace budou požadavky obecné, funkční a technologické. Dále vymezíme hranice Android aplikace MedAsistent jakožto komponenty v celém měřicím systému. Popíšeme zvolenou architekturu aplikace včetně jejího výběru a zmíníme strukturu REST API a zdrojů dat.

6.1. Obecné požadavky

Aplikace je určena pro dotykový panel (tablet s operačním systémem Android), který je v místnosti vrchní sestry a trvale zobrazuje naměřené teploty. Má mít tedy funkci nástěnky (dashboard[18]), jejímž cílem je přehledná vizualizace teploty pacienta nebo lůžka.

Pro běh nástroje je potřeba instalace aplikace na vhodném chytrém zařízení běžícím na operačním systému Android. Předpokladem je častá aktualizace dat, řádově v desítkách sekund až jednotkách minut. Pro zařízení bude zajištěno stálé internetové připojení a stálý přísun elektrické energie.

Nástroj se v současnosti nebude integrovat do žádného stávajícího systému. Zdrojová data jsou využívána ze zdrojové databáze a generovaných datových souborů webové aplikace. Je žádoucí, aby server obsluhoval zároveň jak webovou aplikaci, tak aplikaci pro Android. Celkové zásahy do zmíněných částí jsou minimální. Z hlediska rozsahu budeme implementovat aplikaci pro Android, nutné změny v kódu obsluhující server a případně databázi.

Aplikace je primárně určena zdravotním sestrám. Proto zahrnujeme funkcionalitu, která odráží jejich požadavky na funkce, k nimž mají nastavená uživatelská práva i ve webové aplikaci. Pro ovládání není potřeba žádných specifických znalostí. Ovládání by tedy mělo být jednoduché a uživatelské prostředí intuitivní a přehledné v českém jazyce.

[18] Dashboard, z angl.„nástěnka“, se využívá u produktů, které mají za cíl integrovat informace z více složek do jednotného zobrazení. Ve světě internetu je používána nejčastěji jako označení místa v uživatelském rozhraní, na kterém se zobrazují klíčové informace, nejnovější aktualizace, zprávy a upozornění. [45]

6.2. Funkční požadavky

Hlavní strana

Nejdůležitější funkcí aplikace je přehledné zobrazení současné teploty všech pacientů založených v systému včetně jejich identifikace (například jméno, číslo lůžka[19]). Průběhy teplot se zobrazí do grafu a aktuální teplota číselnou hodnotou.

Forma hlavní strany je preferovaná jako sada dlaždic, kde každá dlaždice obsahuje záznam jednoho pacienta. Velikost dlaždice by měla být minimální (včetně velikosti vnějších i vnitřních okrajů) tak, aby se na obrazovku vešlo co nejvíce pacientů. Dále je vhodné implementovat zvýraznění a řazení dlaždic tak, aby byla na první pohled patrná teplota mimo stanovený rozsah a aktuální stav senzoru.

Je-li aktuální naměřená teplota vysoká (tedy nad zadaný limit, například nad 37 °C), pak se dlaždice zbarví příslušnou barvou. Dalším požadavkem na informativní zobrazení v dlaždici je zobrazení ikony v dlaždici, a to v případě detekování slabé baterie v senzoru a v případě výpadku měřených dat.

Vzhledem k funkci dlouhodobého zobrazení hlavní stránky je nutné zajistit, aby aplikace nepřetržitě běžela včetně podsvícení displeje a nepřecházela do úsporného režimu.

Každá dlaždice může obsahovat i jednoduchý malý graf, který ale nebude interaktivní, pouze má ukazovat krátkou historii naměřených teplot za posledních 30 minut s vyznačením oblasti hodnot, kterou zdravotnický personál považuje za vysokou teplotu. Při klepnutí na dlaždici pacienta se přejde na stránku detailu pacienta, tedy zobrazí se velký graf v režimu na celou obrazovku.

Přesun na další dlaždice by mělo být řešeno vertikálním scrollováním. Vzhledem k tomu, že se jedná o hlavní stránku aplikace, je potřeba zajistit přístup na další funkce aplikace. Ty jsou přístupné výhradně přes kontextové menu nebo jinou prostorově nevýraznou komponentu.

[19] V následujícím textu budeme používat k identifikaci pacienta jeho jméno, nicméně jej lze nahradit číslem lůžka nebo jakýmkoliv jiným identifikátorem, jehož volba závisí na příslušném zdravotnickém zařízení.

Detail pacienta

Detail pacienta je přístupný klepnutím (neboli z angl. „tapnutí“) na dlaždici konkrétního pacienta na hlavní stránce. Z této stránky je možnost návratu na hlavní stranu, který by měl být také řízen časovačem na jednu minutu. Při nečinnosti uživatele automaticky přesměruje zpět na hlavní stranu. Hlavní komponentou je velký graf sloužící k vyhledávání historie naměřených hodnot, vyjma jména pacienta další detaily zobrazovat nebude.

Založení a smazání pacienta

Další požadovanou funkcí je založení pacienta s identifikací nezávislou na systému, tedy pacienta bude možné identifikovat, jak si dané zdravotnické středisko určí. Pro ukončení měření se bude pacient ručně mazat ze systému. Prohlížení historie naměřených teplot již ukončených měření není potřeba implementovat, k tomuto účelu bude nadále sloužit webová aplikace.

6.3. Bezpečnostní požadavky

Z důvodu využití důvěrných dat a nutnosti ochrany osobních údajů pacientů je potřeba zajistit přístup do systému výhradně přes přihlašovací systém.

Pro přihlášení je vyžadováno uživatelské jméno a heslo, pro registraci navíc e-mail. Heslo musí být z důvodu dostatečného zabezpečení alespoň 8 znaků dlouhé. Vzhledem k využití aplikace je potřeba zajistit dlouhodobé přihlášení do systému bez automatického odhlášení samotným systémem.

Odhlášení je tedy možné provést pouze ručně.

Systém rolí není třeba aplikovat z již výše zmíněného důvodu (aplikace je určena výhradně zdravotním sestrám), vyjma kontroly uživatele, že má k dané operaci dostatečná práva.

Stažení aplikace nebude veřejně dostupné.

6.4. Technická specifikace

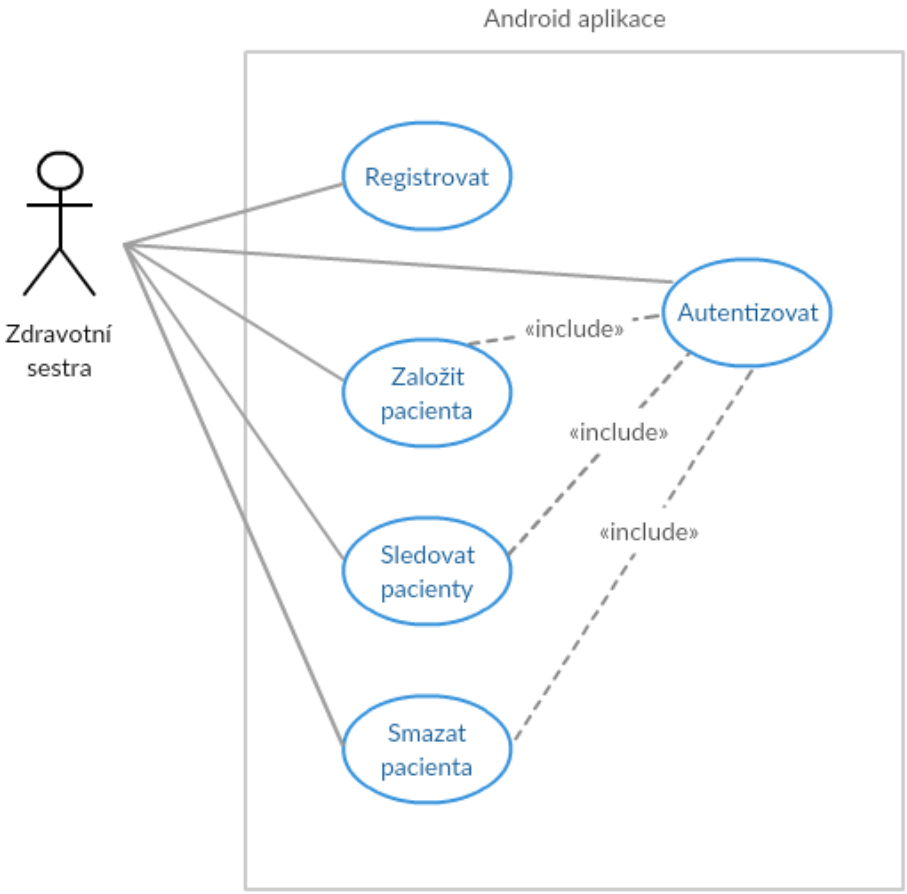
Aplikace je určena pro běh na operačním systému Android, funkční pro verzi 4.3 a vyšší (API 18). Důvodem je stále velké zastoupení využití Android verze 4 (viz tab. 2). Nejsou kladeny konkrétní limity na dobu odezvy systému. Preferované zařízení pro běh aplikace je tablet, pro který je rozhraní optimalizováno.

Verze	Název	API	Distribuce
2.3.3 – 2.3.7	Gingerbread	10	0.8 %
4.0.3 – 4.0.4	Ice Cream Sandwich	15	0.8 %
4.1.x	Jelly Bean	16	3.1 %
4.2.x		17	4.4 %
4.3		18	1.3 %
4.4	KitKat	19	18.1 %
5.0	Lollipop	21	8.2 %
5.1		22	22.6 %
6.0	Marshmallow	23	31.2 %
7.0	Nougat	24	8.9 %
7.1		25	0.6 %

Tabulka 2: Zastoupení verzí OS Android k 5. 6. 2017 [39]

6.5. Případy užití

Obr. č. 6.1 zobrazuje případy užití Android aplikace.



Obrázek 6.1: Případy užití

6.6. Architektura

Při návrhu architektury bylo potřeba se rozhodnout mezi variantami spojení Android aplikace ke vzdálené databázi – přímé připojení (například pomocí JDBC) nebo s využitím Service Oriented Architecture (SOA), která by fungovala jako rozhraní mezi klient-server aplikací.

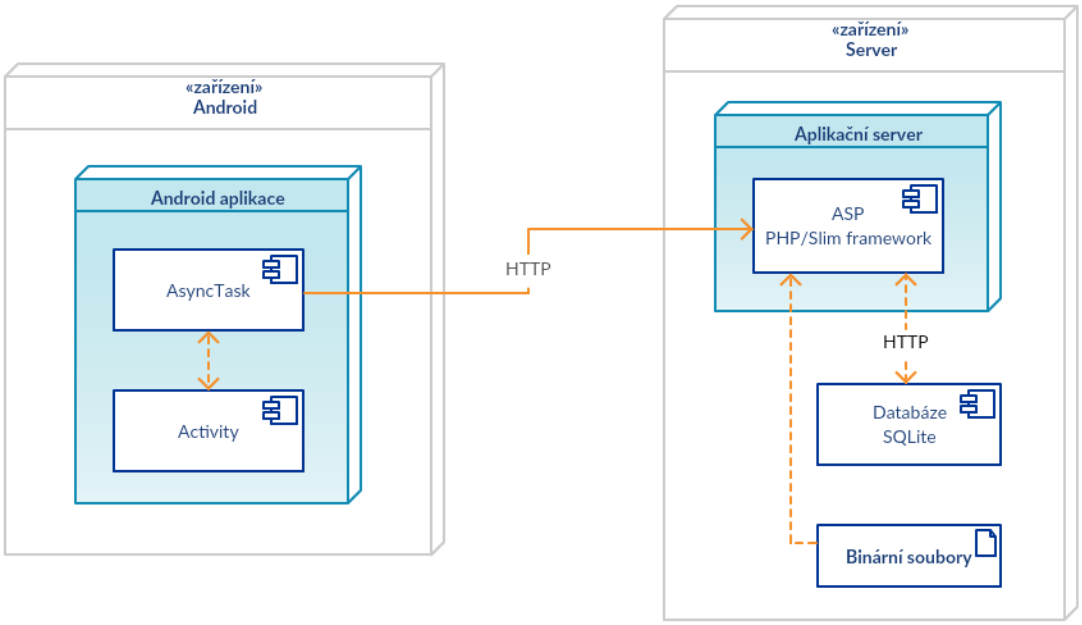
Přestože přímé připojení aplikace k SQLite databázi možné je, tato varianta byla vyřazena z následujících důvodů:

- Aplikace pro Android může být dekompilována a získán tak neautorizovaný přístup do databáze.
- Vzhledem k tomu, že aplikace využívá jako jeden ze zdrojů dat tři různé databáze a připojení do databáze trvá dlouhou dobu, vznikly by problémy v podobě špatné odezvy.
- Dalším důvodem je potřeba časté aktualizace dat podle potřeb uživatele. Buď by muselo být drženo nepřetržité připojení do určité databáze i v případě nevyužití, nebo by naopak muselo být spojení často navazováno.

Jako použitá architektura bylo zvoleno složení ze dvou aplikací, tedy architektura orientovaná na služby. Android aplikace tu funguje jako klient a business logiku obsluhuje server v roli application service provider (ASP). Ten vytváří webové služby, provádí CRUD operace (Create, Read, Update, Delete) a snadno stanovuje zásady pro autentizaci a autorizaci uživatelů. V našem případě je toto řešení vhodnější z mnoha důvodů:

- Umístění v blízkosti zdrojové databáze, z čehož plyne minimalizace odezvy, jelikož mezi klientem a serverem jsou přenášena pouze nutná data.
- Na stejné architektuře již byla implementována webová aplikace a s příslušnými změnami je možné ovládat zároveň webovou i Android aplikaci.
- Z toho plyne i snížení složitosti při nasazení obou systémů.
- Při rozšiřování funkcionality je zvolené řešení též výhodné, jelikož business logika bude implementována jednou a společně pro webovou i Android aplikaci.

Android aplikace tedy funguje jako klient připojující se k webové službě, která přistupuje k jednotlivým databázím a v našem případě ještě k datovým souborům, které jsou uloženy na témže serveru. Jednotlivé komponenty zobrazuje diagram nasazení na obr. 6.2.



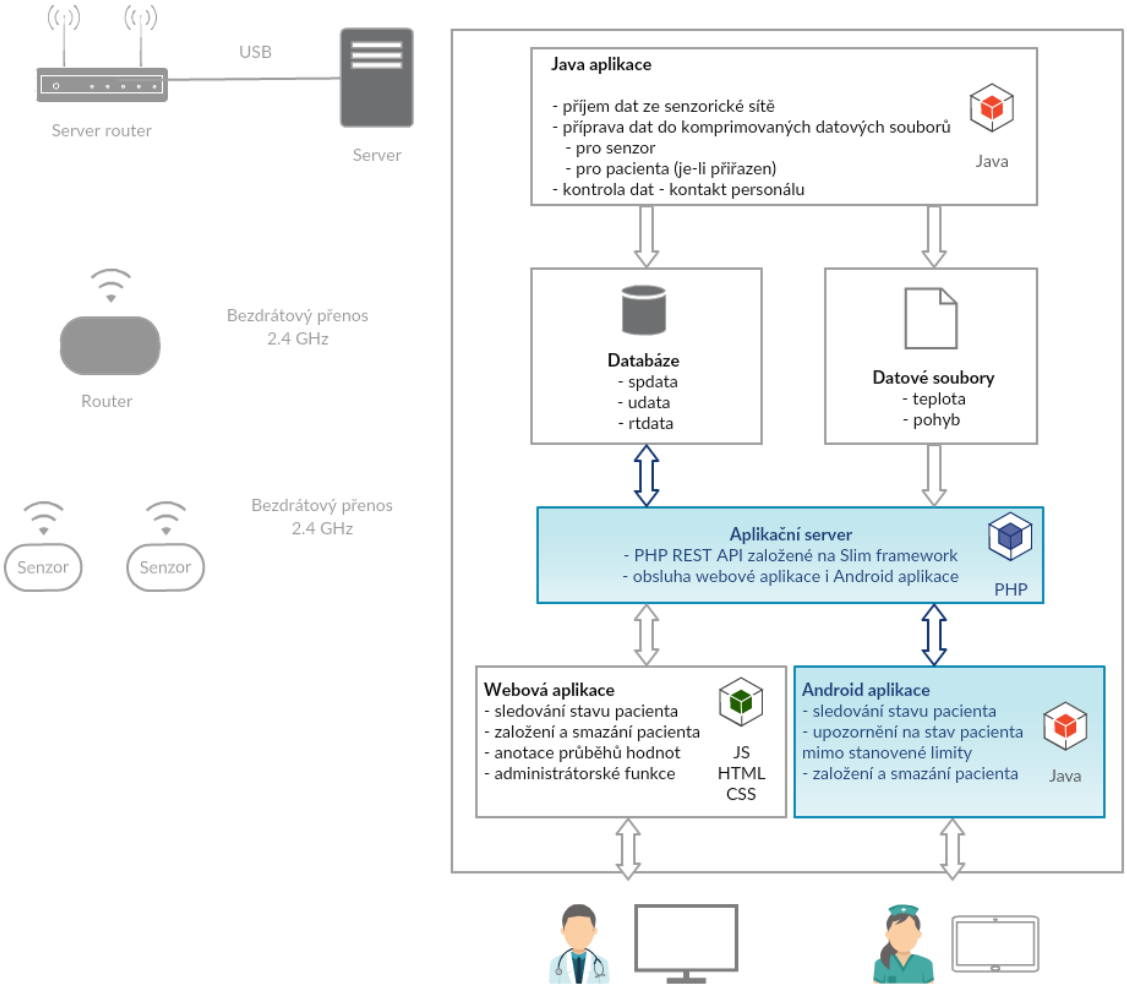
Obrázek 6.2: Diagram nasazení

- Android aplikace je podporovaná od API 9. Pro komunikaci s webovou službou jsou určeny AsyncTask komponenty, které získávají, vkládají nebo vymazávají záznamy z databází.
- Webová služba je implementována v PHP a struktura kódu založena na Slim framework, který dodržuje REST API principy. Jako aplikační sever je používán Apache.
- Sever dále hostuje databáze SQLite a datové binární soubory, které jsou též uloženy fyzicky na serveru.

6.7. Vymezení hranic systému

Předmětem této práce je vytvoření nativní aplikace pro platformu Android, která má obdobnou funkcionalitu jako již implementovaná webová aplikace.

Zařazení této aplikace do senzorického systému je zobrazeno na obr 6.3. Implementace aplikačního serveru není předmětem této práce, nicméně jej bude potřeba upravit tak, aby vyhovoval současnému použití webové i Android aplikace.



Obrázek 6.3: Vymezení Android aplikace v měřicím systému

6.8. REST API

Webová služba je implementována v souladu s REST architekturou za použití Slim framework. Ten obsluhuje všechny HTTP metody (get, post, put, delete), které jsou potřeba pro manipulaci s daty. Data odesílaná serverem jsou ve formátu JSON.

HTTP operace	CRUD operace
GET	CREATE
POST	INSERT
PUT	UPDATE
DELETE	DELETE

Tabulka 3: Srovnání HTTP a CRUD operací

Pro účely Android aplikace zůstane REST API webové aplikace zachováno s drobnými změnami.

REST URI, které bude Android aplikace využívat, zobrazuje tab. 4.

Typ operace	URI	Popis
GET	/patient/active	Získej seznam měřených pacientů/aktivních měření
POST	/patient	Vytvoř pacienta/založ měření
DELETE	/patient/:id	Smaž pacienta s ID zadaným paramentem
GET	/sensor/:voltage	Získej seznam senzorů s napětím baterie menším než zadaný parametr
GET	/sensor/available	Získej seznam dostupných senzorů

Tabulka 4: Poskytované služby pro Android aplikaci

6.9. Data

Jak jsme již zmínili v kap. 4.2, datové zdroje aplikace jsou jednak databázové soubory, a jednak datové binární soubory.

Jako databázový systém je použit SQLite 3, což je malá knihovna vyznačující se absencí serveru a databázového systému ve formě abstrahovaného prostředí. Každá databáze sdružuje veškeré tabulky do jednoho souboru, jež je uložený na serveru.

Aplikace využívá stejně jako webová aplikace 3 databáze: spdata, rtdata a udata.

- spdata pro účely pro informace o měřeních pacientů a přiřazení senzorů
- rtdata pro získání konkrétních informaci o jednotlivých senzorech zavedených do systému
- udata pro účely registrace a přihlášení

Databáze udata, respektive její jediná tabulka users se dočkala změny, a to v podobě přidání atributu password_salt, který souvisí se zabezpečením hesla. Důvod jeho přidání si vysvětlíme v dalším textu. Android aplikace nebude využívat tabulku annotations v databázi apdata vzhledem k tomu, že použití anotací je určeno uživatelům s rolí admin (tedy například lékařům), kteří by měli využívat spíše obsáhlejší webovou aplikaci. Současnou strukturu databází zobrazuje obr. 6.4.

Datové binární soubory (viz kap. 4.2.2), jejich struktura a adresářová struktura uložení zůstává zcela nezměněna.

Oba typy datových zdrojů budou využívané zároveň webovou aplikací i aplikací pro Android.

Obrázek 6.4: Struktura databázového systému

7. Implementace

V této kapitole se budeme zabývat implementací Android aplikace MedAsistent a dalším informacím, které se implementace týkají. Vysvětlíme a odůvodníme postupy a způsoby, které byly vybrány pro vytvoření cílové aplikace. Představíme si její nejdůležitější části, ve kterých je obsažena funkčnost specifikovaná v předešlé kapitole. Pro bližší seznámení s kódem je vhodná prohlídka zdrojových kódů, které jsou součástí přiloženého CD.

7.1. Vývojové prostředí

Jelikož vyvíjíme aplikaci pro platformu Android, budeme potřebovat vývojovou sadu Android SDK [40] a vývojové prostředí. Zvolili jsme Android Studio verze 2.3.3 jako oficiální IDE pro vývoj aplikací na platformu Android. Na oficiálních stránkách[20] lze stáhnout celý balík obsahující vše potřebné k vývoji aplikace.

Součástí Android studia je build systém Gradle, který sestaví výslednou aplikaci na základě dvou konfiguračních souborů. První je umístěn v `<projekt>/app/build.gradle`, který je specifický pro daný modul (například aplikace nebo knihovna). Obsahuje všechny externí závislosti a konfigurace výsledné aplikace, například:

- `minSdkVersion`: minimální verze API, na kterém bude možné aplikaci spustit (v našem případě nastaveno na 15),
- `compileSdkVersion`: API verze, proti které se aplikace sestavuje (v našem případě nastaveno na 23). Zde je doporučeno nastavit vždy nejvyšší možný level, aby aplikace podporovala funkčnosti z nejnovější verze Androidu. V případě nastavení na nižší verzi a zároveň spuštění na zařízení s API vyšší verze, by byla použita funkčnost zavedená vyšší verzí a mohlo by dojít k chybě.

Projekt může obsahovat více nezávislých modulů a pro identifikaci a konfiguraci Gradle je k dispozici druhý konfigurační soubor v `<projekt>/build.gradle`.

[20] https://developer.android.com/studio/index.html

7.2. Uživatelské rozhraní

Základním stavebním prvkem Android aplikace je aktivita – programová část vykonávající kód hlavního vlákna programu a obsluhující události. V naší aplikaci je každé aktivitě přiřazen XML soubor typu Layout, který definuje rozložení prvků uživatelského rozhraní dané aktivity. Společně tedy tvoří vzhled obrazovky aplikace.

MainActivity jako jediná obsahuje Fragment, takže kromě Layout na nejvyšší úrovni reprezentující aktivitu obsahuje ještě Layout fragmentu. Fragment pak obsahuje GridLayout, který uspořádává jednotlivé prvky (items, v naší aplikaci dlaždice pacienta) do mřížky.

Ostatní aktivity mají Layout velmi jednoduchý, obsahují nativní komponenty jako TextView, EditView, Button, Spinner, případně LineChart jako komponentu knihovny MPAndroidChart (více v dalším textu).

Všechny důležité prvky Views jsou inicializovány v příslušném souboru obsahující Layout a ty jsou na základě přiřazení unikátního identifikátoru dynamicky naplňovány daty za běhu aplikace.

Primárním účelem je použití aplikace na tabletu. Aktivity, které neobsahují formulář (MainActivity a DetailActivity), mají nastavenou orientaci na landscape. Tím je zajištěn požadavek co největšího množství dlaždic pacientů na hlavní stránce a přehledný graf historie naměřených teplot v detailu pacienta.

7.3. Významné programové části

7.3.1. Activity

Aktivita představuje jednu obrazovku umožňující nějakou funkčnost a interakci mezi uživatelským rozhraním a uživatelem. Veškeré třídy Activity jsou obsaženy ve stejnojmenném adresáři. V následujícím textu je výčet použitých aktivit a stručný popis jejich funkčnosti.

- LoginActivity.java

Jak název activity napovídá, tato aktivita slouží k přihlášení uživatele do aplikace. Je první stránkou, která se zobrazí při spuštění aplikace, a obsahuje jednoduchý formulář dotazující se na uživatelské jméno (login) a heslo. Její kód odesílá přihlašovací údaje aplikačnímu serveru a v případě nalezení uživatele je vrácen objekt k údaji uživatele včetně serverem vygenerovaného session ID sloužící k autentifikaci při dalších dotazů. Poté jsou údaje uživatele zavedeny do lokální databáze pro účely udržení síťového spojení mezi klientem a serverem pro daného uživatele. V případě úspěšného přihlášení je uživatel přesměrován na hlavní stránku (MainActivity.java).

Dále se ve spodní části nachází odkaz k přechodu na RegisterActivity.java.

- RegisterActivity.java

Další activity je obrazovka registrace, do které se vstupuje přes přihlašovací obrazovku. Jedná se opět o registrační formulář, dotazující se na uživatelské jméno, uživatelský e-mail a heslo. Po validaci vložených údajů jsou údaje odeslány serveru, jenž uživatele přidá do databáze.

- MainActivity.java

MainActivity představuje funkčně nejdůležitější část aplikace, a to hlavní stránku. Ta má funkci zmíněného dashboardu (nástěnky), u níž je předpokládáno, že bude zobrazena po většinu času běhu aplikace. Při spuštění vytvoří fragment a připravuje veškerá data (stažení JSON objektu, parsování, vytvoření instancí tříd, řazení), kterými následně volaný PatientAdapter.java stránku naplní. Dále se stará

o pravidelnou aktualizaci fragmentu v intervalu nastaveném v konfiguračním souboru.

Na hlavní stránce je též dostupné množství dalších ovládacích prvků aplikace. Přes kontextové menu lze přejít k založení pacienta, odhlášení z aplikace a přes klepnutí na dlaždici pacienta přejít na DetailActivity.

- DetailActivity.java

Jedná se o obrazovku, jejíž komponentou je interaktivní graf naplněný naměřenými daty. Aktivita je zodpovědná za jeho přípravu a naplnění daty, podobně jako MainActivity s tím rozdílem, že zde se zobrazují historická data až po současnost. Dále obsahuje tlačítko fungující jako pokyn ke smazání pacienta, tedy ukončení jeho měření.

Při vytvoření aktivity se spustí časovač, jenž má za úkol při neaktivitě uživatele po uplynutí nastaveného času přesměrovat zpět na hlavní stránku, aby opět aplikace zaujala svou primární funkci nástěnky.

- NewPatientActivity.java

Aktivita umožňující založení měření pacienta. Obsahuje opět jednoduchý formulář pro vyplnění jména pacienta (nebo jakékoliv jiné identifikace), seznam senzorů zavedených do systému a zároveň dostupných pro nové měření a volitelný popis k senzoru. Po úspěšném vyplnění je pokyn odeslán serveru a pacient přidán do databáze.

Tato aktivita též obsahuje časovač, který při delší neaktivitě uživatele přesměruje na hlavní stránku.

7.3.2. Adapter

Jediným použitým adaptérem je PatientAdapter.java, třída vzniklá implementací rozhraní BaseAdapter. Má funkci mostu mezi položkami GridView a fragmentem v MainActivity jakožto zdroji dat položek.

7.3.3. AsyncTask

Pro komunikaci klienta a serveru je použita generická abstraktní třída AsyncTask, která zajišťuje běh kódu na jiném vlákně. Kód použitých metod je obdobný, liší se ale typem HTTP operace, datovými typy, množstvím a obsahem zadaných parametrů. Do metody vždy vstupuje zadané URI požadavku a session ID uživatele (respektive klienta), které se uloží do hlavičky požadavku. Požadavek se vykoná příkazem `execute`. Do výstupu metody se uloží status požadavku (použitý dále pro odchycení chyb) a odpověď, která v případě úspěšného vykonání představuje JSON objekt s požadovanými daty.

- ReadPatientTask.java

Operace GET nad databází pacientů, která vrací všechny pacienty s aktivním měřením v systému.

```
public class ReadPatientTask extends AsyncTask<String, Void,
        List<String>> {

    @Override
    protected List<String> doInBackground(String... params) {

        int connectionStatusCode;
        String serverResponse;
        List<String> taskResponse = new ArrayList<>();

        try {
            URL url = new URL(params[0]);
            String session_id = params[1];

            HttpClient client = new DefaultHttpClient();
            HttpGet get = new HttpGet(String.valueOf(url));
            get.setHeader("Cookie", "PHPSESSID=" + session_id + ";");

            HttpResponse httpResponse = client.execute(get);
            connectionStatusCode =
               httpResponse.getStatusLine().getStatusCode();
            HttpEntity entity = httpResponse.getEntity();
            serverResponse = EntityUtils.toString(entity);

            taskResponse.add(Integer.toString(connectionStatusCode));
            taskResponse.add(serverResponse);

        } catch (IOException e) {
            System.out.println("ReadPatientTask: IOException
               caught!");
            e.printStackTrace();
        }
        return taskResponse;
    }
}
```

- ReadBinaryTask.java

Slouží k načtení dat z binárních souborů uložených na serveru a následném převedení jednotlivých bytů do pole typu String.

- AddPatientTask.java

Operace POST vytváří nové měření a ukládá jej do databáze.

- DeletePatientTask.java

Operace DELETE slouží k ukončení měření pacienta.

- ReadSensorAvailableTask.java

Operace GET pro získání senzorů zavedených v systému a dostupných pro nová měření.

- ReadSensorInfo.java

Operace GET provádí načtení obsahu databáze senzorů zavedených v systému. Pro účely aplikace je využívána zejména informace k identifikaci slabé baterie nebo jiného chybového stavu senzoru.

7.3.4. Vizualizace do grafů

Pro potřeby vizualizace naměřených dat je vhodné vykreslení do liniového grafu. Zvolená knihovna musí splňovat následující požadavky:

- Open-source knihovna
- Interaktivita
- Implementace pro Android, aby byla zajištěna bezproblémová interakce na chytrém zařízení
- Plynulé vykreslení velkého množství hodnot (v řádu tisíců)
- API umožňující úpravy vzhledu a funkčnosti

Jednou z možností bylo použití stejné knihovny (dygraphs.js) jako ve webové aplikaci vložením HTML komponenty do aplikace, nicméně má nedostatečnou interaktivitu při použití na dotykovém displeji a zároveň neposkytuje API pro použití na chytrém zařízení.

Po analýze dostupných nativních Android knihoven byla vybrána knihovna MPAndroidChart v3.0.1 [41].

7.3.5. Kontrola internetového připojení

Vzhledem k tomu, že funkčnost aplikace je zcela závislá na funkčním internetovém připojení, je potřeba zajistit odpovídající chování v případě jeho výpadku. Kontrola dostupnosti probíhá dvěma způsoby:

Každá aktivita před voláním metod, které spolupracují se serverem, kontroluje, zda je připojení dostupné, tedy zda probíhá výměna datových toků. To zajišťuje zachycení výpadku jak z důvodu vypnuté wi-fi/mobilních dat, tak výpadek datových toků mimo Android zařízení.

```
public static boolean isNetworkAvailable(Context context) {
    try {
        if (context != null) {
            ConnectivityManager connectivityManager =
                (ConnectivityManager)Context
                .getSystemService(CONNECTIVITY_SERVICE);
            NetworkInfo activeNetworkInfo =
                connectivityManager.getActiveNetworkInfo();
            return activeNetworkInfo != null &&
                activeNetworkInfo.isConnected();

        } else {
            return false;
        }
    } catch (Exception e) {
        Log.e(NetworkStateChangeReceiver.class.getName(),
                e.getMessage());
        return false;
    }
}
```

Další komponenta představuje receiver, který naslouchá v celé aplikaci změně stavu internetového připojení (NetworkStateChangeReceiver.java). Každá aktivita registruje broadcast receiver tak, že vytvoří intent a implementuje metodu onReceive. Ta se spuští vždy, když se broadcast receiver aktivuje.

```
IntentFilter intentFilter = new
IntentFilter(NetworkStateChangeReceiver.NETWORK_AVAILABLE_ACTION);

LocalBroadcastManager.getInstance(this).registerReceiver(new
BroadcastReceiver() {
    @Override
    public void onReceive(Context context, Intent intent) {
        //…
    }
}, intentFilter);
```

V případě, že je detekováno neaktivní internetové připojení, zobrazí se ve spodní části oznámení, které přetrvá do obnovení připojení.

7.3.6. Konfigurace aplikace

Pro souhrnný přehled nastavované konfigurace v aplikaci slouží soubor AppConfig.java. Obsahuje jednak URL serveru a jednak odvozená URI veškerých požadavků volaných aplikací. V případě, že dojde k migraci serveru, stačí na straně klienta změnit jednu hodnotu URL, která automaticky modifikuje i ostatní adresy. Dále obsahuje číselné hodnoty sloužící k personalizaci klientské aplikace, například:

- limit teploty, která je považována již za vysokou pro účely zvýraznění,
- požadovaná doba aktualizace dat na hlavní stránce,
- množství dat (doba) zobrazované v jednotlivých dlaždicích na hlavní stránce, aj.

7.3.7. Dialogy

V operačním systému Android je mnoho způsobů, kterými lze uživatele informovat o nějaké události. V případě krátké informativní zprávy je použit bublinový dialog Toast (obr. 7.1), který po chvíli automaticky zmizí.

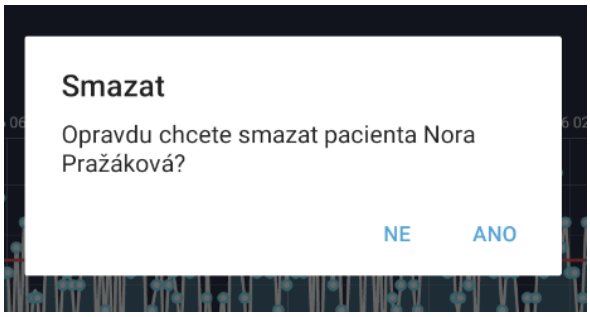
Obrázek 7.1: Ukázka dialogu Toast

Pro účely důležitějších informativních zpráv, tedy v našem případě upozornění na neaktivní internetové připojení, je zvolena komponenta Snackbar (obr. 7.2). Ta se vždy zobrazí ve spodní části obrazovky, lze u ní zajistit perzistentní zobrazení a v případě potřeby lze přidat ovládací prvky.

Obrázek 7.2: Ukázka dialogu Snackbar

Další použitou třídou dialogů je AlertDialog (obr. 7.3), který vyžaduje asistenci uživatele. Nezmizí tedy automaticky, ale je zobrazen, dokud nedojde ke kliknutí

na jedno z tlačítek. AlertDialog je vhodný pro důležité zprávy, u kterých potřebujeme potvrzení, že si je uživatel skutečně přečetl.



Obrázek 7.3: Ukázka dialogu AlertDialog

7.4. Manifest

K tomu, aby mohla aplikace přistupovat k různým zdrojům a informacím v zařízení, musí se v souboru AndroidManifest.xml definovat požadovaná oprávnění (permission).

Uživatel je pak v průběhu instalace aplikace informován o tom, jaká oprávnění aplikace vyžaduje, a pro dokončení instalace musí požadovaná oprávnění potvrdit.

```
<uses-permission
        android:name="android.permission.INTERNET" />
<uses-permission
        android:name="android.permission.ACCESS_NETWORK_STATE" />
<uses-permission
        android:name="android.permission.ACCESS_WIFI_STATE" />
```

Všechna oprávnění souvisí s tím, že celá aplikace je závislá na komunikaci s aplikačním serverem. První oprávnění je tedy přístup k internetu potřebný pro stahování, ukládání a mazání dat. Další oprávnění souvisí se získáním informace o stavu připojení k internetu. Pokud aplikace zjistí, že zařízení není připojené k internetu, neprovádí žádné požadavky směrem k serveru.

Další důležitou komponentou registrovanou v Manifestu, která souvisí s dostupností internetového připojení, je Broadcast Receiver.

Přidaný Intent filter znamená, že kdykoliv intent zachytí událost `android.net.conn.CONNECTIVITY_CHANGE` (Android je odpojen od internetového připojení a naopak) a Receiver tento intent obslouží.

```
<receiver
    android:name=".helper.NetworkStateChangeReceiver"
    android:exported="false">
    <intent-filter>
        <action android:name="android.net.conn.CONNECTIVITY_CHANGE"/>
    </intent-filter>
</receiver>
```

V Manifestu jsou deklarovány všechny aktivity v aplikaci a nastaveny základní parametry jako například text hlavičky či orientace obrazovky dané aktivity.

7.5. Autentizace

Před vykonáním každého požadavku klienta směrem k serveru probíhá kontrola autentizace pomocí serverem vygenerovaného session ID po úspěšném přihlášení. Jelikož platnost session ID je dána serverem a vzhledem k použití aplikace na dotykové panely, je doba trvání platnosti session prodloužena, aby se zabránilo častému odhlašování uživatele.

Změnou oproti webové aplikaci prošla šifrovací funkce pro ukládání hesla. Původní MD5 není považována za bezpečnou z důvodu možné kolize[21] otisků. Pro šifrování hesel byla vybrána kombinace SHA-1, náhodný řetězec a kódování algoritmem Base64. SHA-1 je hašovací algoritmus, nelze jej tedy dekódovat.

Při registraci posílá klient parametry (uživatelské jméno, e-mail a heslo) jako součást HTTP požadavku na server. Nejprve je vygenerován náhodný řetězec salt, se kterým je zřetězeno originální heslo. Tento řetězec je zahašován algoritmem SHA-1. Výsledný řetězec je poté zakódován metodou Base64. Každé heslo využívá v databázi dva sloupce password a password_salt, kde první je šifrované heslo a salt vygenerovaný řetězec.

Uživatelské jméno a heslo se tedy posílá v hašované podobě. Nejedná se však o kryptografické zabezpečení přihlašovacích údajů. Použití této metody předpokládá komunikaci na zabezpečeném kanálu mezi klientem a serverem.

[21] Je-li u hash algoritmu možné vytvořit tzv. kolizi, tedy uměle vytvořit dva stejné otisky, není algoritmus dále bezpečný.

7.6. Autorizace

Aby bylo zamezeno neoprávněnému přístupu k datům na serveru, probíhá kontrola v serverovém API. Autentizace probíhá jak na straně klienta, tak na straně serveru. Dále se kontrolují dostatečná práva uživatele na vykonání operace. V případě odmítnutí přístupu se PHP operace nevykoná a vrátí chybu.

Tento způsob zabezpečení je zvolen zejména z důvodu kompatibility serverového API s původní webovou aplikací. Následující kód kontroluje přihlášení a práva uživatele na straně serveru.

```
function authorize($role) {
    return function () use($role) {
        // Get the Slim framework object
        $app = Slim::getInstance ();

        // Check to see if the user is logged in at all
        if (! empty ( $_SESSION ['user'] )) {
            // Validate role to make sure they can access the route
            if ($_SESSION ['user'] ['role'] >= $role) {
                return true;
            }
            else {
                // User is logged in, but doesn't have permissions
                $app->halt ( 403, '403: Pristup odmitnut.');
            }
        } else {
            // User is not logged in at all, return a 401
            $app->halt ( 401, '401: Nejste prihlasen.' );
        }
    };
}
```

7.7. Distribuce aplikace

Vzhledem k tomu, že aplikace bude součástí celého měřicího systému a bez účasti všech komponent, je její použití zcela limitováno, základním požadavkem je její neveřejná distribuce. Standardní publikování aplikace skrze veřejnou část služby Google Play není tedy vhodné.

Pro distribuci této aplikace je žádoucí zvolit místo na publikování privátních aplikací, ať to bude například Google Play pro publikaci privátních aplikací či Appkilt[22]. Výhodou těchto služeb je možnost zajištění upozornění na nové aktualizace a možnost různých distribucí nových verzí.

[22] http://www.appkilt.com/

8. Ověření funkcionality

Tato kapitola se zabývá testováním aplikace, popisuje zařízení, na nichž byla aplikace testována, a způsoby, kterými byla testována v průběhu a po dokončení vývoje. Budou zde uvedeny testované funkce a ukázány snímky obrazovek aplikace na testovaném tabletu. Dále zhodnotíme implementovanou aplikaci a navrhneme možná rozšíření.

Aplikace byla vyvíjena v prostředí Android Studio. Obecně je možné aplikaci spouštět na všech zařízeních od verze systému 4.0.3 (API 15), což znamená funkcionalitu na 99 % zařízeních na trhu. Aplikace byla kompilována cílovým API 25, tedy v době vývoje aktuálně nejnovější.

8.1. Testovaná zařízení

Cílovou skupinou použitých zařízení byly tablety a mobilní telefony. Pro účely testování aplikace byla použita tři testovaná fyzická zařízení (jeden tablet a dva telefony) a dvě emulovaná zařízení (jeden tablet a jeden telefon).

Fyzická zařízení

- ASUS Zenpad S 8 – verze systému 5.0 Lollipop (API 21), úhlopříčka displeje 8“, rozlišení displeje 1536 x 2048 bodů
- Lenovo S60-a – verze systému 5.0 Lollipop (API 21), úhlopříčka displeje 5“, rozlišení displeje 1280 x 720 bodů
- Samsung Galaxy S3 – verze systému 4.3 JellyBean (API 18), úhlopříčka displeje 4.8“, rozlišení displeje 1280 x 720 bodů

Emulovaná zařízení

- Pixel C – verze systému 7.1 Nougat (API 25), úhlopříčka displeje 9.94“, rozlišení displeje 2560 x 1800 bodů
- Nexus One – verze systému 4.0 IceCreamSandwich (API 15), úhlopříčka displeje 3.7“, rozlišení displeje 800 x 480 bodů

Výběr emulovaných zařízení je motivován testováním na nejnižším a nejvyšším deklarovaném API a na rozdílných velikostech a rozlišeních displejů.

Na všech uvedených zařízeních proběhla úspěšně instalace aplikace, spuštění aplikace, otestování veškerých funkcí a odinstalace aplikace.

8.2. Manuální testování

V průběhu posledních fází vývoje a po jeho skončení byla aplikace na výše uvedených fyzických zařízeních testována manuálně. Manuální testování bylo prováděno osobami, které neznají detaily implementace, a to na jejich vlastních zařízeních s OS Android. V úvodu byla osoba provádějící testování seznámena, jaký je účel této aplikace a jaké má funkce.

Při testování byly zaznamenány nedostatky a pády aplikace v různých situacích, jejichž příčiny se podařilo identifikovat a následně opravit. Manuálním testováním byla ověřena funkcionalita důležitých částí aplikace:

- registrace, přihlášení, odhlášení
- hlavní stránka a vykreslovaných dat
- přechod na detail pacienta a interakce s grafem
- založení a smazání pacienta

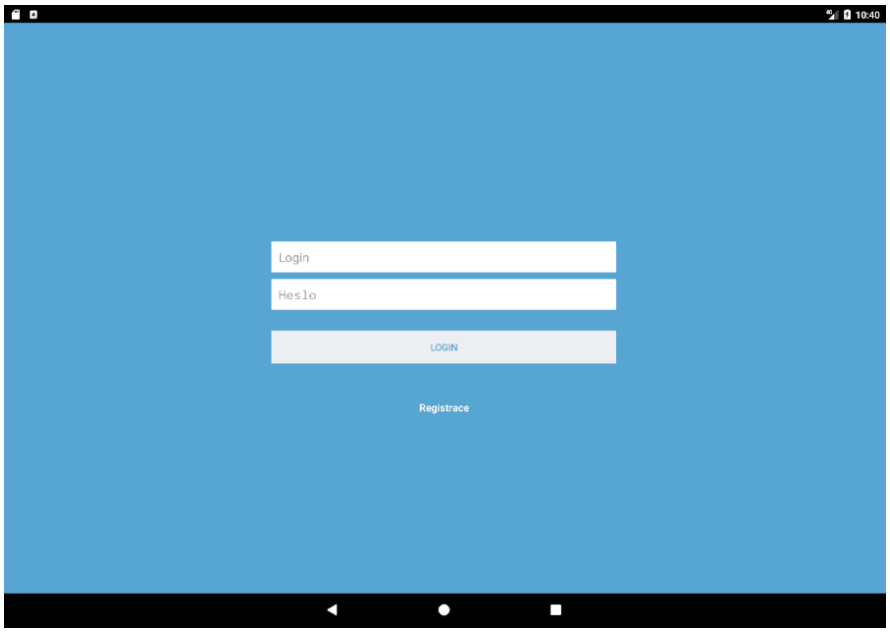
Jako největší potenciální nedostatek byla spatřována rychlost při načítání a parsování dat na hlavní stránku a detail pacienta. Například pokud u detailu pacienta budeme předpokládat, že aktivita načítá hodnoty za uplynulé 3 dny při frekvenci měření jedenkrát za minutu, pak načítáme a parsujeme 4320 dvojic hodnot. Paradoxně osoby provádějící testování k rychlosti žádné výhrady neměli.

Aplikace běží na testovaných zařízeních plynule, pomalejší reakce byly zaznamenány u zařízení Samsung Galaxy S3 (model představený na počátku roku 2012), jehož hardware je již poměrně zastaralý a téměř všechny aplikace představují na tomto zařízení problém.

8.3. Ukázky aplikace

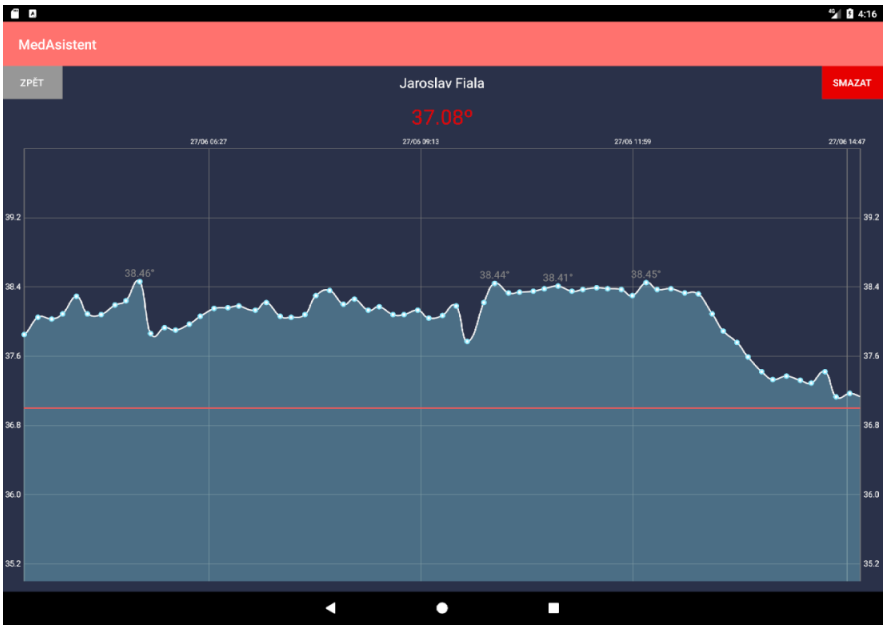
Na obrázcích 8.1 až 8.3 jsou pro ilustraci zobrazeny některé snímky z aplikace.

Na prvním snímku je přihlašovací obrazovka. Na druhém snímku je zobrazena nejdůležitější hlavní stránka s dlaždicemi pacientů, která zobrazuje aktuální teplotu pacienta a krátký vývoj měření za posledních 30 minut. Na dalších snímcích je detail pacienta a formulář pro založení nového pacienta. Tyto snímky jsou pořízeny z emulovaného zařízení Pixel C, jehož specifikace je popsána v kap. 8.1. Více snímků z aplikace lze nalézt v uživatelské příručce v příloze C.

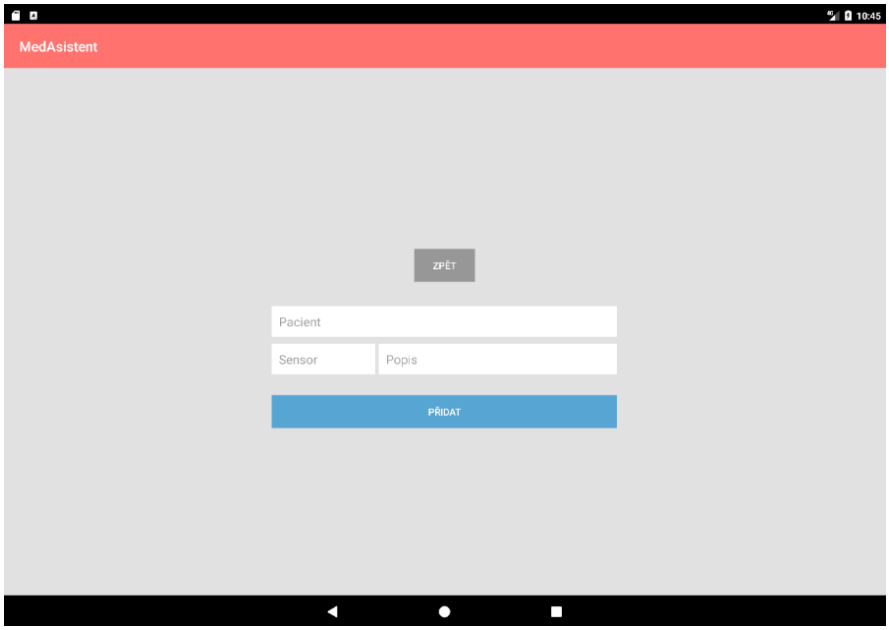


Obrázek 8.1: Přihlašovací stránka

Obrázek 8.2: Hlavní stránka



Obrázek 8.3: Detail měření pacienta



Obrázek 8.4: Založení měření pacienta

8.4. Zhodnocení

Jak již bylo uvedeno v předchozím textu, aplikace se věnuje vizualizaci měřených teplotních dat. V následujícím výčtu budou jednotlivé funkce zmíněny:

Hlavní stránka

- Zobrazení aktuálních teplot pacientů v přehledných dlaždicích, které jsou řazeny podle hodnoty teploty
- Kontrola a upozornění stavu baterie senzoru
- Pravidelná aktualizace dat

Správa pacientů

- Založení nového pacienta a přiřazení dostupného senzoru
- Ukončení aktivního měření pacienta

Detail pacienta

- Procházení historie naměřených teplot pacienta v interaktivním grafu

Obecné funkce

- Registrace nového uživatele do systému
- Přihlášení uživatele, bez kterého není umožněna hlavní funkčnost aplikace
- Kontrola aktivního internetového připojení v celé aplikaci
- Informační dialogy pro oznámení uživateli o výsledku operací

Veškeré funkce byly otestovány na uvedených referenčních zařízeních a také na zařízeních uživatelů, kteří používáním aplikace napomohli ke komplexnějšímu testování. Implementované funkce odpovídají návrhu (viz kap. 6).

8.5. Možnosti rozšíření

Aplikace v její první verzi splňuje základní požadavky pro vizualizaci naměřených dat. Existuje však potenciál k dalšímu rozvoji.

Napojení na IS zdravotnického zařízení včetně integrace uživatelských účtů

Zobrazení více informací o stavu senzorů:

- Lokalita výskytu senzoru

  Aplikace by mohla zobrazovat souhrnnou tabulku senzorů zavedených do systému. Zobrazení lokality senzoru může být užitečné při případné ztrátě senzoru či získání lokality pacienta v případě nevolnosti, kolapsu apod. Naopak je třeba zmínit, že pohyb nositele senzoru by potenciálně mohl být sledován. Senzor nicméně umožňuje sledování lokality pouze v oblastech nasazení routerů, tedy například na oddělení nemocnice, kde má pacient v průběhu hospitalizace omezený pohyb.

- Stáří přijatých dat

  Je vhodné v případě, že je senzor nefunkční.

Detekce sejmutí senzoru pro účely automatického ukončení měření

Například pohybuje-li se teplota v měřeném intervalu 120 minut na nízkých hodnotách (pokojová teplota), indikuje to, že senzor není umístěn na lidském těle, tedy neprobíhá měření, a je možné automaticky měření ukončit.

Možnost volby rozložení hlavní stránky

Vzhledem k použití fragmentů je vhodná možnost rozložení, například mřížka dlaždic, tabulka pacientů (seznam, který lze abecedně řadit), tabulka v levé části a mřížka v části pravé.

Alarm manager

Zařízení běžící na OS Android má potenciál v široké škále upozornění uživatele na daný stav. Například lze implementovat zvukovou notifikaci v případě, že pacient překročil daný limit pro horečku (nad 38 °C) nebo alarm v případě vysoké horečky (nad 40 °C).

9. Závěr

Téma tohoto projektu je zajímavé zejména pro jeho využití ve zdravotnickém průmyslu, který prochází masivním technologickým rozvojem. Monitorování tělesné teploty u pacientů může přinést stálé sledování vývoje teploty a umožnit tak včasnou reakci na vývoj zdraví člověka.

Nové technologie – výzkumné centrum Západočeské univerzity v Plzni v současnosti vyvíjí systém bezdrátových senzorů, které měří a zaznamenávají hodnoty tělesné teploty. Cílem vývoje tohoto projektu je zcela nahradit nutnost používání klasických teploměrů pro měření tělesné teploty a ručního zaznamenávání hodnot do papírových archů. Přináší tak jistou formu modernizace, jejímž výsledkem je zvýšení komfortu jak pro pacienty, tak pro zdravotnický personál. Dochází ke zrychlení informovanosti personálu o stavu pacienta, a tím i ke zkvalitnění zdravotní péče. Součástí současné verze senzorického systému je i webová aplikace, která sloužila primárně pro detailní rozbor naměřených dat. Její použití ale v praxi není vhodné z důvodu odlišných požadavků na funkčnost aplikace.

Z testování systému uskutečněného v Klatovské nemocnici, a.s. vyplynulo, že optimálním způsobem, jak přehledně zpřístupnit aktuální teploty pacientů personálu, je pomocí dotykového panelu, který bude umístěn v místnosti vrchní sestry, případně na dalších vhodných místech. Hlavním zadáním této práce bylo vyvinout aplikaci pro dotykový panel na operačním systému Android, která přehledně zpřístupní informace o tělesné teplotě pacientů a stavu teplotních senzorů. Aplikace byla realizována na základě požadavků plynoucích z informací od zdravotnického personálu.

Úvodem této práce bylo poskytnuto teoretické zázemí k problematice měření tělesné teploty a jeho zaznamenávání. Bylo popsáno, jak měření vyvíjenými teplotními senzory probíhá a jaké jsou předpoklady ke správnému měření. Z existující analýzy akceptovatelnosti senzorů pacienty a hodnocení systému zdravotnickým personálem byla určena požadovaná funkčnost na aplikaci pro dotykový panel.

Cílem této práce bylo vybrat a implementovat zvolenou funkčnost současné webové aplikace sloužící k vizualizaci měřených dat pro mobilní zařízení na platformě Android s primárním zaměřením na tablet. Mobilní aplikace MedAsistent má sloužit zdravotnickému personálu, zejména zdravotním sestrám. Základními funkčními požadavky bylo přehledné znázornění seznamu pacientů a jejich aktuálních teplot, historie naměřených dat konkrétního pacienta a možnost založení a smazání pacienta. Vzhledem k potřebě anonymizace pacientových dat bylo nutné zajistit dostupnost dat pouze oprávněným osobám formou přihlašovacího systému.

Vybrané funkce aplikace pro vizualizaci naměřených dat pro platformu Android byly navrženy a implementovány. Architektura aplikace je zvolena tak, aby bylo možné současně používat webovou aplikaci i Android aplikaci. Toto řešení je výhodné pro případ budoucích změn v obou systémech.

Na závěr byly uvedeny možnosti dalšího rozšíření funkcionality aplikace.

Zadání práce bylo splněno. Aplikace bude v další fázi testování nasazena a dle nových požadavků plynoucích z použití bude upravena či rozšířena.

Seznam obrázků

Seznam tabulek

Reference

1. Sledování fyziologických funkcí. *WikiSkripta.* [Online] [Citace: 11. 15 2015.] http://www.wikiskripta.eu/index.php/Sledov%C3%A1n%C3%AD_fyziologick%C3%BDch_funkc%C3%AD.

2. Správné měření. *Hartmann.* [Online] Hartmann. [Citace: 15. 11 2015.] http://thermoval.cz/spravne-mereni.html.

3. **Matoušková, Eva.** *Měření tělesné teploty.* [Online] 28. 9 2009. [Citace: 15. 11 2015.] http://www.szsmb.cz/admin/upload/sekce_materialy/T%C4%9Blesn%C3%A1_teplota.pdf.

4. Přesné měření tělesné teploty. *Rehabilitace.info.* [Online] Rehabilitace.info, 10. 12 2013. [Citace: 20. 11 2015.] http://www.rehabilitace.info/zdravotni/presne-mereni-telesne-teploty/.

5. **Allegaert, Karel, a další, a další.** *Tympanic, Infrared Skin, and Temporal Artery Scan Thermometers Compared with Rectal Measurement in Children: A Real-Life Assessment.* [https://www.ncbi.nlm.nih.gov/pmc/articles/PMC4008772/]

6. **Pompei, Marybeth.** Temperature Assessment via the Temporal Artery: Validation of a New Method. [Online] 1999. [Citace: 19. 03 2016.] http://www.quickmedical.com/downloads/exergen-temperature-assessment-via-temporal-artery.pdf.

7. BioDigital: Human visualization platform. [Online] [Citace: 9. 10 2016.] https://human.biodigital.com/.

8. Jak změřit dítěti teplotu. *JáRodič.* [Online] 19. 7 2010. [Citace: 25. 11 2015.] http://www.jarodic.cz/cz/jak-zmerit-diteti-teplotu.php.

9. **Vytejčková, Renata.** Sledování a hodnocení fyziologických funkcí. [Online] [Citace: 21. 11 2015.] http://nas.lf3.cuni.cz/materialy/CNSOP1/sledovani%20a%20hodnoceni%20fyziologickych%20funkci(5087ce78cdced).pdf.

10. **Vytejčková, Renata, a další, a další.** *Ošetřovatelské postupy v péči o nemocné II: Speciální část.* Praha : Grada Publishing, a.s., 2013. 978-80-247-3420-0.

11. Introducing the 3M™ SpotOn™ Temperature Monitoring System. [Online] [Citace: 03. 04 2017.] http://multimedia.3m.com/mws/media/878163O/spoton-system-brochure.pdf.

12. *Exploratory Method-Comparison Evaluation of a Disposable Non-Invasive Zero Heat Flow Thermometry System.* **Eshraghi, Y. a Sessler, D. I.** : American Society of Anesthesiologists Annual Meeting, 2012. A63.

13. Smartphone. *Wikipedie.* [Online] 24. 03 2016. [Citace: 29. 03 2016.] https://cs.wikipedia.org/wiki/Smartphone.

14. Top 8 Mobile Operating Systems. *StatCounter.* [Online] 2017. [Citace: 25. 06 2017.] http://gs.statcounter.com/os-market-share/mobile/worldwide/#monthly-201706-201706-bar.

15. Top 7 Tablet OSs from Mar 2015 to Mar 2016. *StatCounter.* [Online] 2016. [Citace: 30. 04 2016.] http://gs.statcounter.com/#tablet-os-ww-monthly-201503-201603.

16. Number of available applications in the Google Play Store from December 2009 to February 2016. *The Statistics Portal.* [Online] 2016. [Citace: 30. 04 2016.] http://www.statista.com/statistics/266210/number-of-available-applications-in-the-google-play-store/.

17. Number of available apps in the Apple App Store from July 2008 to June 2015. *The Statistics Portal.* [Online] 2016. [Citace: 04. 30 2016.] http://www.statista.com/statistics/263795/number-of-available-apps-in-the-apple-app-store/.

18. Vývoj Windows Phone aplikací. [Online] VIP Trust s.r.o., 08. 01 2016. [Citace: 20. 03 2016.] http://viptrust.cz/vyvoj-windows-phone-aplikaci/.

19. Number of apps available in leading app stores as of July 2015. *The Statistics Portal.* [Online] 2016. [Citace: 15. 03 2016.] http://www.statista.com/statistics/276623/number-of-apps-available-in-leading-app-stores/.

20. **Vrecková, Helena.** *Smartphony umožní některá základní vyšetření.* [http://vtm.e15.cz/smartphony-umozni-nektera-zakladni-vysetreni] : E15 VTM.

21. Smith-root.com. [Online] Smith Root Inc., 2017. [Citace: 11. 05 2017.] http://store.smith-root.com/catalog/biomeme-two3-device-p-455.html.

22. **Brát, Jiří.** *Návrh a implementace webové aplikace pro monitorování pacientů.* Západočeská univerzita v Plzni. Plzeň : autor neznámý, 2015. Bakalářská práce.

23. **Sircus, Dr.** *Low Body Temperature Symptoms and Causes – And How to Treat It.* [http://drsircus.com/wp-content/uploads/2015/01/lowbody2.png] : DrSircus.com, 21. 1 2015.

24. *Measuring Body Temperature Wirelessly: Patients' and Personnel's Acceptance in a Hospital Environment. Manuscript submitted for publication.* **Mertl, Jiří a Tolar, David.** 2017.

25. **Kalbach, Jim.** The First Responsive Design Website: Audi (circa 2002). *EXPERIENCING INFORMATION.* [Online] 22. 07 2012. [Citace: 11. 03 2016.] https://experiencinginformation.wordpress.com/2012/07/22/the-first-responsive-design-website-audi-circa-2002/.

26. RESPONSIVE WEB DESIGN. *Promoting Webs.* [Online] [Citace: 10. 03 2016.] http://www.promotingwebs.com/wp-content/uploads/2015/02/responsive-design-businesses.png.

27. **Developers, Google.** Installable Web Apps (100 Days of Google Dev). [Online] 26. 06 2015. [Citace: 30. 03 2016.] https://www.youtube.com/watch?v=N1Bdu7ukN40.

28. Multikanálová budoucnost: jaké mobilní aplikace budete vytvářet vy? *ITBIZ.* [Online] 24. 11 2013. [Citace: 02. 04 2016.] http://www.itbiz.cz/clanky/multikanalova-budoucnost-jake-mobilni-aplikace-budete-vytvaret-vy.

29. **Václavík, Jan.** Jak vyvíjet mobilní aplikace. *Jan Václavík.* [Online] 04. 06 2015. [Citace: 02. 04 2016.] http://janvaclavik.cz/jak-vyvijet-mobilni-aplikace/.

30. Vyvíjíme hybridní aplikace v Ionicu: Úvod a instalace. *zdroják.cz.* [Online] 20. 07 2015. [Citace: 02. 04 2016.] https://www.zdrojak.cz/clanky/vyvijime-hybridni-aplikace-v-ionicu/.

31. jQuery mobile. [Online] [Citace: 11. 03 2016.] https://jquerymobile.com/.

32. AngularJS. [Online] [Citace: 11. 03 2016.] https://angularjs.org/.

33. ionic. [Online] [Citace: 11. 03 2016.] http://ionicframework.com/.

34. **Eason, Jamal.** An update on Eclipse Android Developer Tools. *Android Developers Blog.* [Online] 26. 6 2015. [Citace: 20. 1 2016.] https://android-developers.googleblog.com/2015/06/an-update-on-eclipse-android-developer.html.

35. Android Studio. [Online] [Citace: 23. 10 2016.] https://developer.android.com/studio/intro/migrate.html#manifest-settings.

36. **Hlavík, Jiří.** *3. díl - Android programování - Životní cyklus a nový projekt.* [https://www.itnetwork.cz/java/android/tutorial-programovani-pro-android-v-jave-zivotni-cyklus-a-novy-projekt] : ITnetwork.cz, 2016.

37. **TutorialsPoint.** *Android - UI Layouts.* [https://www.tutorialspoint.com/android/android_user_interface_layouts.htm] 2017.

38. Declaring Permissions. *Android Developers.* [Online] [Citace: 10. 03 2016.] http://developer.android.com/training/permissions/declaring.html.

39. Dashboards: Platform Versions. *Android Developer.* [Online] [Citace: 5. 6 2017.] https://developer.android.com/about/dashboards/index.html .

40. *Android Developers.* [Online] [Citace: 10. 04 2016.] http://developer.android.com/sdk/index.html.

41. **Jahoda, Philipp.** *MPAndroidChart.* [https://github.com/PhilJay/MPAndroidChart/wiki] 2017.

42. **Jurášková, Olga a Horňák, Pavel.** *Velký slovník marketinových komunikací.* Praha : Grada Publishing, a.s., 2012. ISBN 978-80-247-4354-7.

43. **Vrbacký, Jakub.** Technologie mobilního internetu – od CSD po LTE Advanced (vědecké okénko). *Mobilizujeme.* [Online] 12. 02 2012. [Citace: 02. 03 2016.] http://mobilizujeme.cz/clanky/technologie-mobilniho-internetu-od-csd-po-lte-advanced-vedecke-okenko/.

44. **Kreidl, Marcel a Šmíd, Radislav.** *Technická diagnostika - senzory, metody, analýza signálu.* Praha : BEN - technická literatura, 2006. ISBN 80-7300-158-6.

45. **IT SLOVNÍK.cz.** *Dashboard.* [https://it-slovnik.cz/pojem/dashboard/?utm_source=cp&utm_medium=link&utm_campaign

Použité diagramy a schémata jsou vlastního zdroje a byly vytvořené v online nástroji Creately[23]. =cp] 05. 02 2017.

[23] https://creately.com/

Přílohy

A Odpovědi dotazníkového šetření v rámci studie

V této části jsou uvedeny odpovědi dotazníkového šetření v rámci studie „Measuring Body Temperature Wirelessly: Patients' and Personnel's Acceptance in a Hospital Environment" [24], jejíž shrnutí se nachází v kap. 4.5.

Jak hodnotíte obecný přínos bezdrátového teploměru?		**Myslíte si, že bezdrátový teploměr by mohl být zaveden v nemocničním prostředí v některé formě?**	
Rozhodně dobrý nápad	20	Rozhodně ano	19
Spíše dobrý nápad	5	Spíše ano	6
Ani dobrý ani špatný nápad	1	Spíše ne	0
Spíše špatný nápad	1	Rozhodně ne	1
Rozhodně špatný nápad	0	Nevím	1

Jakou zkušenost jste získal během nošení bezdrátového teploměru?		**Myslíte si, že bezdrátový teploměr by mohl být zaveden v nemocničním prostředí v aktuální formě?**	
Dobré	13	Rozhodně ano	9
Spíše dobré	9	Spíše ano	8
Ani dobré ani špatné	4	Spíše ne	6
Spíše špatné	1	Rozhodně ne	2
Špatné	0	Nevím	2

Zaznamenávali jste bezdrátový teploměr v době, kdy jste ho nosili?		**Vadil vám bezdrátový teploměr během spánku?**	
Ano, neustále	1	Vůbec jsem jej nezaznamenal	19
Často	3	Občas překážel	8
Z času na čas	12	Často překážel	0
Vůbec ne	11	Nemohl/a jsem spát	0

Pokud nosíte brýle, jak jste je nosili s teploměrem současně?		**Byla pro vás vhodná fyzická realizace (velikost a tvar)?**	
Bez problémů	10	Naprosto vhodná	9
Překážel minimálně	8	Spíše vhodná	7
Občas překážel	5	Ani vhodná ani nevhodná	7
Měl/a jsem problém s nošením současně	0	Spíše vhodná	3
Nenosím brýle	4	Zcela nevhodná	0

Jak jste se cítil/a během nošení teploměru?		**Zaznamenali jste, že vaše okolí vnímá, že nosíte teploměr?**	
Příjemně	5	Ano, stále	1
Spíše příjemně	8	Někdy	10
Ani příjemně, ani nepříjemně	13	Vůbec ne	16
Spíše nepříjemně	0		
Nepříjemně	1		

Zaznamenal/a jste možnost, že jste byli neustále monitorován/a ?		**Odlepoval se teploměr, když jste ho nosil/a?**	
Ano a vadí mi to	0	Ano, pravidelně	8
Ano, ale nevadí mi to	10	Jen jednou	13
Vůbec ne	17	Vůbec ne	5
		Nevím/Nepamatuji si	1

B Instalační příručka

Aplikaci lze spouštět na zařízení s operačním systémem Android verze 4.0.3 až 7.1. Pro instalaci aplikace je potřeba stáhnout do zařízení instalační soubor APK z adresáře /Aplikace/APK. Pro dokončení instalace je třeba mít v nastavení zařízení povolenou instalaci z neznámých zdrojů. Cesta k povolení je následující:

Nastavení > *Zabezpečení* > *Neznámé zdroje* >

> *Povolit instalaci aplikací z důvěryhodných i neznámých zdrojů*

Poté stačí potvrdit požadovaná oprávnění aplikace a instalaci dokončit.

Pro potřeby testování aplikace je vytvořeno testovací prostředí na serveru, které generuje teplotní data do binárních souborů stejným způsobem, jako by probíhalo reálné měření. Stejně tak je na testovacím serveru k dispozici serverová část, lze tedy otestovat veškerou funkčnost aplikace.

Přiložené CD dále obsahuje kompletní projekt pro IDE Android Studio s veškerými zdrojovými kódy v adresáři /Aplikace/Projekt, generovaná dokumentace javadoc je v adresáři /Aplikace/javadoc.

Kód serverové části je k nahlédnutí v adresáři /Aplikace/Server.

C Uživatelská příručka

Pro účely testování aplikace jsou generovaná data měření.

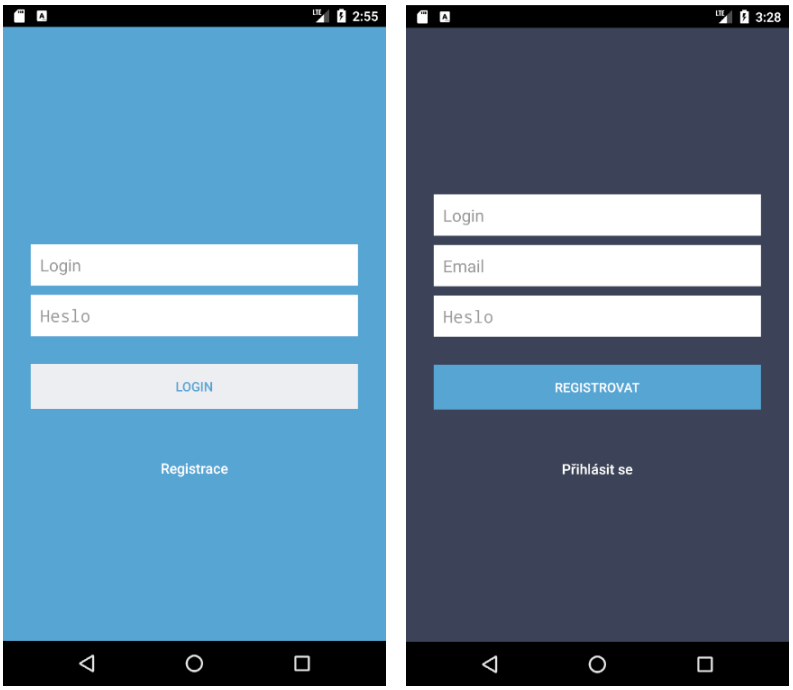
Přihlášení, výchozí obrazovka

Po spuštění aplikace se zobrazí přihlašovací obrazovka s poli pro přihlášení (obr. 10.1).

Pro přihlášení je nutné vyplnit uživatelské jméno, heslo a potvrdit tlačítkem PŘIHLÁSIT SE. V případě velké velikosti systémové klávesnice na obrazovce je umožněno přejít na následující pole klávesou ENTER, případně kliknutím do volného prostoru klávesnici zavřít.

Registrace

Nemá-li uživatel účet, lze jej založit na stránce Registrace (obr. 10.2), ze které se přechází z přihlašovací stránky po kliknutí na odkaz REGISTRACE. Pro zaregistrování nového uživatelského účtu je potřeba vyplnit uživatelské jméno délka alespoň 4 znaky, platnou e-mailovou adresu a heslo délky alespoň 8 znaků. Po úspěšné registraci je uživatel přesměrován zpět na přihlašovací stránku, kde se může ihned přihlásit do aplikace.



Obrázky 10.1 a 10.2: Přihlašovací stránka a Registrační stránka

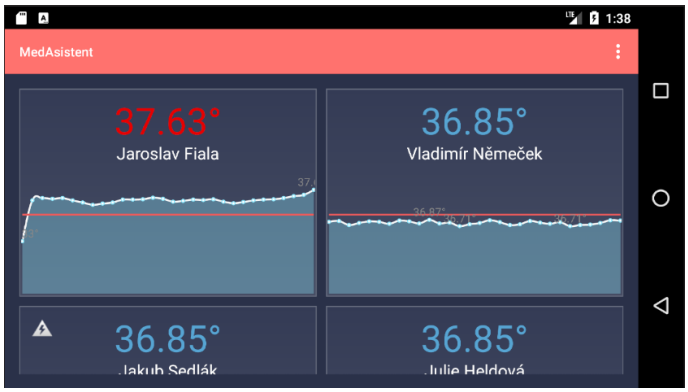
Hlavní stránka

Hlavní stránka (obr 10.3) se skládá z dlaždic jednotlivých pacientů, které jsou řazeny sestupně dle aktuálně nejvyšší naměřené teploty.

Součástí každé dlaždice jsou následující komponenty:

- Jméno pacienta (nebo jiná identifikace)
- Aktuální naměřená teplota, která je zbarvena modře v případě, že hodnota teploty je v mezích normální tělesné teploty, nebo červeně, je-li teplota vyšší
- Graf vývoje naměřených teplot za posledních 30 minut s vyznačením limitu vysoké teploty
- Informativní ikona, která se zobrazí v případě, že senzor, přiřazený pacientovi, má vybitou baterii

Hlavní stránka se aktualizuje každé dvě minuty a je neustále podsvícená, tedy nepřechází do úsporného režimu. Výchozí nastavení orientace obrazovky je landscape, tedy na šířku.

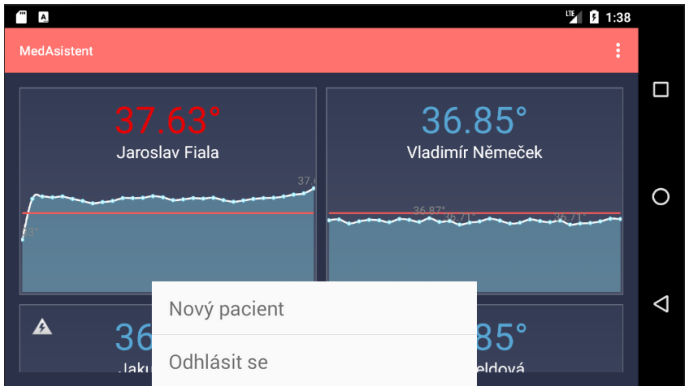


Obrázek 10.3: Hlavní stránka

Navigační menu

Hlavní stránka obsahuje systémové navigační menu (obr. 10.4), které lze zobrazit několika rozdílnými způsoby závisející na výrobci a modelu chytrého zařízení.

- Kliknutím na ikonu ⋮ v pravém horním rohu horní lišty (obr. 10.3)
- Kliknutím na levé spodní tlačítko (může být hardwarové i softwarové) z tzv. overview buttons



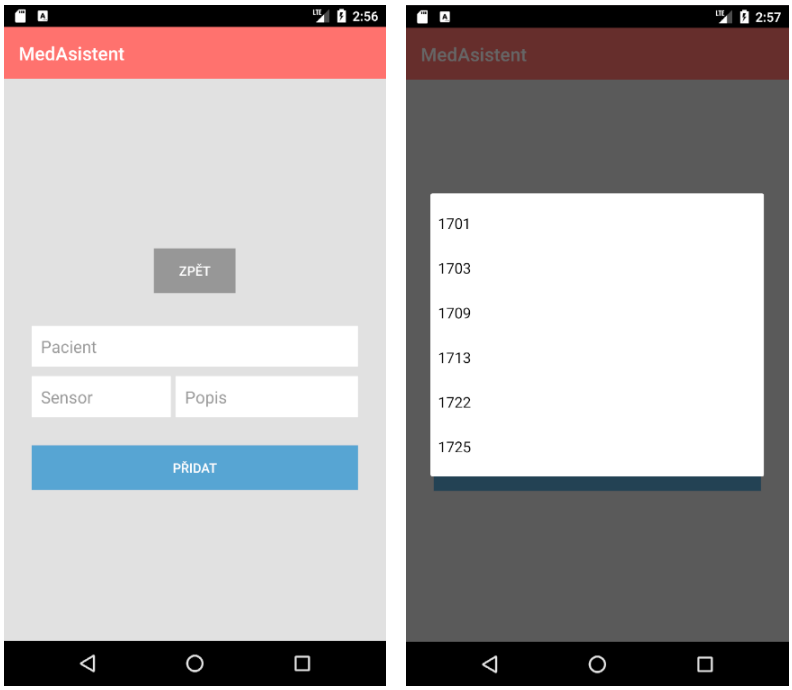
Obrázek 10.4: Systémové navigační menu

Založení pacienta

Založení nového měření pacienta (obr. 10.5) je dostupné přes navigační menu. Pro úspěšné založení je potřeba vyplnit jméno pacienta (nebo jakoukoliv zvolenou identifikaci), vybrat ze seznamu přiřazený senzor (obr 10.6) a kliknout na tlačítko PŘIDAT. Volitelně lze též vyplnit popisek senzoru nebo poznámku k měření. Je-li detekována neaktivita uživatele po dobu 2 minut, zobrazí se po uplynutí této doby automaticky hlavní stránka.

V případě, že seznam neobsahuje požadovaný senzor, může tak být ze dvou důvodů. Buď je senzor využíván k měření teploty jiného pacienta, je tedy nutné stávající měření nejprve ukončit a uvolnit senzor pro nové měření, nebo daný senzor vůbec není zaveden do systému, v tom případě doporučujeme kontaktovat administrátora.

V případě, že seznam nenabízí žádný dostupný senzor, může to být způsobeno dvěma důvody. Prvním může být neaktivní internetového připojení, zkontrolujte tedy jeho dostupnost. Druhým důvodem je, že všechny senzory, zavedené do systému, jsou využívány pro jiná měření.



Obrázky 10.5 a 10.6: Založení pacienta a výběr senzoru

Detail měření

Detail měření pacienta (obr. 10.7) je přístupný po klepnutí na dlaždici pacienta, nacházející se na hlavní stránce. Zobrazí se velký interaktivní graf, v čemž lze prohlížet zvolené období měření. Graf lze přibližovat a oddalovat pomocí dotyku dvou prstů. Výchozí nastavení zobrazuje historii dat po dobu 3 dnů. Je-li detekována neaktivita uživatele po dobu jedné minuty, dojde k automatickému přesměrování na hlavní stránku. Výchozí nastavení orientace obrazovky je landscape, tedy na šířku.

Obrázek 10.7: Detail měření pacienta

Smazání pacienta

Na stránce detailu pacienta lze pacienta smazat, respektive měření ukončit, a to stiskem tlačítka SMAZAT. Poté je uživatel dotázán na potvrzení ukončení. Po úspěšném smazání pacienta je uživatel přesměrován na hlavní stránku.

Odhlášení uživatele

Ruční odhlášení uživatele je přístupné přes navigační menu (obr. 10.4). Po odhlášení je uživatel přesměrován na stránku přihlášení.

Reakce na chyby

- Nedostupné internetové připojení

Funkčnost této aplikace je zcela závislá na aktivním internetovém připojení. V případě, že dojde k jeho výpadku, zobrazí se ve spodní části obrazovky upozornění na neaktivní internetové připojení (obr. 10.8). Zároveň nelze vykonávat veškeré operace (založení, smazání pacienta, apod.) a obrazovka hlavní stránky bude prázdná, jelikož se nepodaří aktualizovat data.

Obrázek 10.8: Dialog nedostupného internetového připojení

- Vypršení relace přihlášení

V případě, že vyprší relace přihlášení, uživatel je přesměrován na přihlašovací obrazovku

- Chyby ze strany serveru

V případě chyby ze strany serveru se zobrazí dialogové okno s výpisem chyby.

D Obsah přiloženého CD

Přiložené CD obsahuje tři hlavní adresáře: Aplikace (kód a spustitelná aplikace klientské, serverová část), Poster a Text (vlastní text diplomové práce):

Aplikace

/APK – instalační soubor APK pro Android zařízení

/Javadoc – generovaná dokumentace projektu v Android Studio

/Projekt – projekt aplikace v Android Studio

/Server – zdrojové kódy serverové části

Poster

- Benesova_Renata_2017.pdf
- Benesova_Renata_2017.pub

Text

- DP_Benesova_Renata.pdf