Slovenská Technická Univerzita v Bratislave
Fakulta Informatiky a Informačných Technológií
Ilkovičova 3, 812 19 Bratislava

Monitorovanie a vyhodnocovanie fyziologických procesov človeka [StresMonitor]

Dokumentácia k inžinierskemu dielu

Vedúci tímu: Ing. Katarína Jelemenská, PhD.
Členovia tímu: Bc. Tomáš Bako, Bc. Martin Baláž, Bc. Matúš Brandýs, Bc. Patrik Husár, Bc. Patrik Krupa, Bc. Matúš Matula, Bc. Kristián Ostatník, Bc. Peter Vašek
Akademický rok: 2017/2018

OBSAH

1 ÚVOD

Dokumentácia k inžinierskemu dielu je dokument, ktorý opisuje technickú časť projektu Monitorovanie a vyhodnocovanie fyziologických procesov človeka (StresMonitor). Projekt je vytváraný v rámci predmetu Tímový projekt tímom 23 s názvom StressBusters. Ako samotný názov projektu naznačuje, zaoberáme sa monitorovaním rôznych fyziologických procesov človeka, ktoré následne spracovávame a vizuálne zobrazujeme na rôznych zariadeniach. Správnym vyhodnotením údajov je možné odhaliť stres alebo iné neprirodzené chovania tela. Celková architektúra systému nie je triviálna, preto tento dokument prináša celkový pohľad na inžinierke dielo a odráža náležitosti návrhu a vývoja jednotlivých častí projektu. Jednotlivé kapitoly približujú systém na rôznych úrovniach abstrakcie. V nasledujúcej kapitole sú opísané globálne ciele a ciele pre zimný a letný semester, ktoré sme sa snažili naplniť. Tretia kapitola obsahuje celkový pohľad na systém, jeho architektúru a dátový model. Štvrtá kapitola popisuje jednotlivé moduly systému a to serverovú časť, mobilnú a webovú aplikáciu. Dokument vo viacerých častiach obsahuje aj UML diagramy pre lepší obraz systému.

2 GLOBÁLNE CIELE

Veľa z nás je často nadmerne fyzicky alebo psychicky zaťažených, čo môže viesť k stresu. So stresom sa stretávame v rôznych oblastiach života, či už v školskom, v pracovnom alebo aj rodinnom prostredí. Stres sa často zanedbáva, pričom sa predpokladá, že nadmerné množstvo stresu môže byť príčinou mnohých zdravotných ťažkostí a to aj tých závažnejších, ako sú rakovina, respiračné, či kardiovaskulárne choroby. Preto hlavným cieľom nášho projektu nie je len upozorniť používateľa v prípade veľkých výkyvov hodnôt, ale aj zabrániť negatívnym následkom na zdraví človeka.

Na základe zadania projektu a stretnutí s produktovým vlastníkom sme zhrnuli globálne ciele do niekoľkých bodov, ktoré sa budeme snažiť naplniť do stanoveného termínu, t.j. do konca druhého semestra v prvom ročníku inžinierskeho štúdia. Cieľom projektu je:

- vytvorenie aplikácie inteligentného bezdrôtového biomonitoringu,
- navrhnutie inovatívnych metód na meranie (meranie so spätnou väzbou), prenos a spracovanie nameraných údajov, špecifické pre vybranú aplikáciu,
- implementovanie a experimentálne overenie prototypu aplikácie.

Keďže náš tím je rozsiahly, rozhodli sme sa vytvoriť okrem mobilnej aplikácie aj webovú s podporou serveru, ktorá bude umožňovať používateľovi väčšie možnosti práce s dátami a ich vizualizáciou. Tieto globálne ciele sú definované všeobecne, preto sa ich špecifikovanie nachádza v nasledujúcich podkapitolách na jednotlivé semestre.

2.1 Ciele pre zimný semester

Náš tím sa skladá z 8 členov, ktorí dovtedy spoločne nepracovali na žiadnom väčšom projekte. Z toho dôvodu je potrebné vyriešiť niekoľko organizačných záležitostí, ako výber verziovacieho a manažovacieho nástroja, komunikáciu v tíme alebo rozdelenie úloh členom tímu. Toto sa stalo prvým cieľom pre zimný semester, ktorý sa budeme snažiť vyriešiť čo najskôr počas prvých spoločných stretnutí, aby mohol tím napredovať správnym smerom.

Cieľom projektu nie je pokračovanie a rozšírenie existujúcej aplikácie, ale vytvorenie nového produktu podľa špecifikácie produktového vlastníka. Tým sa projekt stáva netriviálnym a viac rozsiahlym, čo má za následok vynaloženie väčšieho úsilia na analýzu a návrh architektúry systému alebo metód na spracovanie dát.

Z dôvodu rozsiahlej analýzy v zimnom semestri bude kladený menší dôraz na implementáciu. Cieľom bude vytvorenie len základnej kostry aplikácií so základnými funkciami a komunikáciou medzi nimi. Takto vytvorené aplikácie v spojení s analýzou problémovej oblasti, budú tvoriť základ pre ďalší vývoj modulov systému v letnom semestri.

V rámci zimného semestra sme si s produktovým vlastníkom vytýčili tieto ciele:

- vytvorenie webového sídla tímu,
- analyzovanie interpretácie dát zo zariadení,
- navrhnutie funkcií na spracovanie a vyhodnotenie dát,
- navrhnutie dátového modelu a vytvorenie databázy,
- implementácia servera,
    - zber a ukladanie dát do databázy,
    - základné spracovanie a odoslanie dát do aplikácií,
- implementácia mobilnej a webovej aplikácie so základnými funkciami
    - komunikácia so serverom (prenos dát na server a zo servera),
    - registrácia a prihlásenie používateľa,
    - základná vizualizácia dát.

Tieto ciele boli naplnené do konca zimného semestra.

2.2 Ciele pre letný semester

Pre letný semester sme sa s produktovým vlastníkom dohodli, že sa zameriame na vývoj len mobilnej aplikácie a servera. Preto sme si v rámci letného semestra zadefinovali tieto ciele:

- návrh metód na zisťovanie stresu,
- implementácia servera,
    - metóda na zisťovanie stresu,
    - vyhodnotenie odporúčania pri zvýšenom strese,
- implementácia mobilnej aplikácie,
    - komunikácia so zariadením,
    - zobrazenie aktuálnych hodnôt,
    - zobrazenie historických hodnôt,
    - nastavenia aplikácie,
- experimentálne overenie.

Tieto ciele sa podarilo naplniť do konca letného semestra.

3 CELKOVÝ POHĽAD NA SYSTÉM

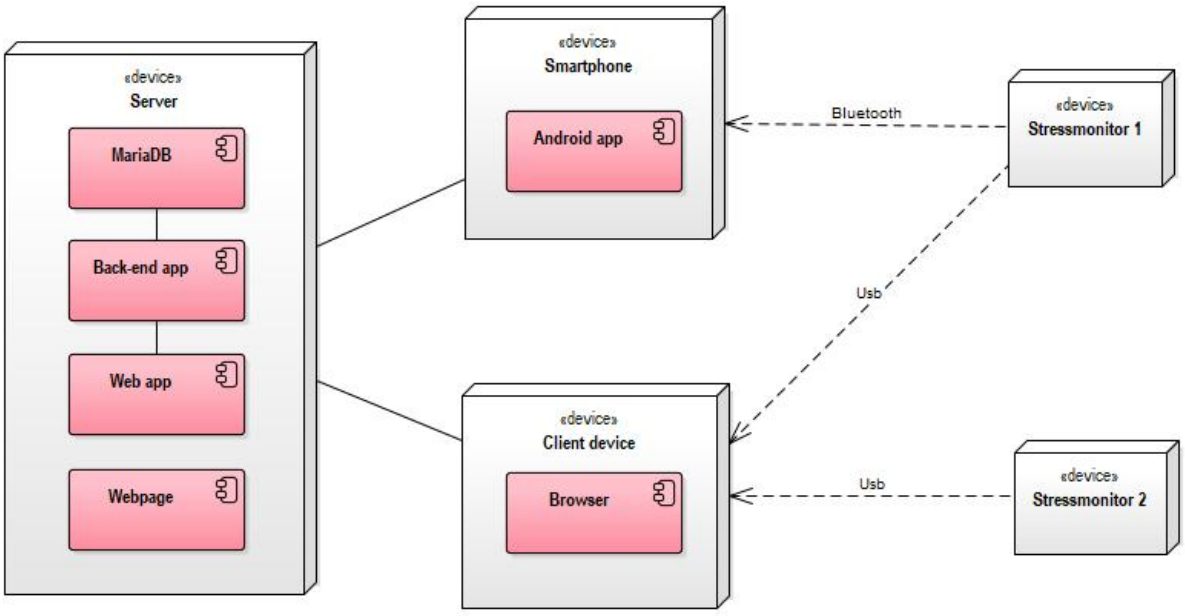
Táto kapitola obsahuje celkový pohľad na technickú stránku nášho projektu. Jednotlivé podkapitoly obsahujú postupne navrhnutú architektúru systému a dátový model. Ďalšie časti budú priebežné pridávané po ich navrhnutí a implementovaní.

3.1 Architektúra systému

Architektúru systému sme sa rozhodli rozdeliť na 3 časti a to na server, mobilnú aplikáciu a webovú aplikáciu. Architektúra predstavuje formu architektúry klient-server.

Meracie zariadenia pre snímanie fyzikálnych charakteristík kože dokážu odoslať dáta rôznym spôsobom, buď pomocou Bluetooth alebo pomocou USB portu. V rámci nášho projektu pracujeme najmä s prvým zariadením, pretože podporuje aj Bluetooth aj USB, ale obsahuje aj viacej snímačov, z ktorých vieme získať dáta. Nameraných dát je veľké množstvo, preto ukladanie a spracovanie na mobilnej alebo webovej aplikácií nie je vhodné. Z tohto dôvodu je server základom nášho systému. Server obsahuje aplikačnú logiku s databázou MariaDB, v ktorej sú uložené všetky informácie o používateľovi a dátach nameraných rôznymi senzormi. Bližší popis databázy a dátového modelu je v nasledujúcej podkapitole.

Komunikácia medzi serverom a aplikáciami je realizovaná pomocou REST API. Klient po prijatí dát zo zariadenia tieto dáta odošle na server, ktorý ich spracuje a uloží do databázy. Opačne, ak chce klient zobraziť namerané dáta, server vyberie podľa požiadavky dáta z databázy, spracuje ich a odošle späť do zariadenia. Získané dáta sú na zariadení vhodne vizualizované a používateľ môže byť informovaný o svojom stave alebo strese. Globálny pohľad na architektúru systému zobrazuje nasledujúci obrázok 1:

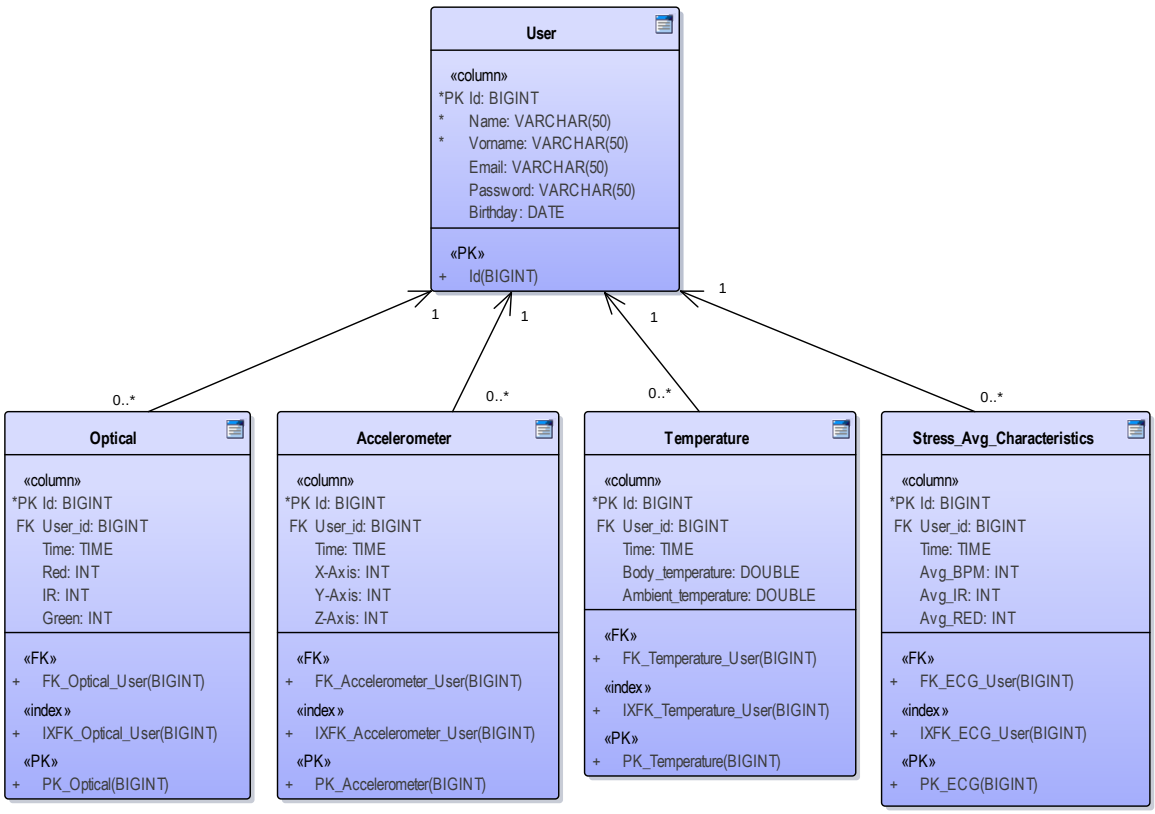


Obrázok 1: Celkový pohľad na systém

3.2 Dátový model

Na serveri sú dáta uložené v databáze MariaDB. Na základe analýzy sme navrhli dátový model, ktorý je zobrazený na obrázku 2. Entity sú medzi sebou poprepájané vzťahmi. Jednotlivé entity sú definované nasledovne:

- **User** – údaje o používateľovi, ktorý sa zaregistroval do systému pomocou aplikácie,
- **Optical** – údaje z optického senzora,
- **Accelerometer** – údaje z akcelerometra,
- **Temperature** – údaje z teplomera,
- **Stress_Avg_Characterics** – priemerné hodnoty pre každého používateľa pomocou, ktorého sa vypočítava stres.



Obrázok 2: Dátový model

4 MODULY SYSTÉMU

Ako už bolo spomínané v predchádzajúcej kapitole, systém je rozdelený na 3 časti: server, mobilná aplikácia a webová aplikácia. V tejto kapitole sú bližšie špecifikované jednotlivé moduly. Ku každému z nich je uvedený popis ohľadom analýzy, návrhu, implementácie a testovania.

4.1 Serverová časť

4.1.1 Analýza

Server, ktorý máme k dispozícií je postavený na virtuálnom stroji. Virtuálny stroj ponúka viacero výhod na rozdiel od fyzického stroja. Jedna z najväčších výhod je, že dokážeme meniť parametre hardvéru daného virtuálneho stroja. V prípade, že je nedostatok pamäte, alebo dochádza k zaplneniu diskového priestoru, je možné tento priestor jednoducho navýšiť. Ďalšia výhoda je schopnosť robiť tzv. „snapshot“, ktorý nám umožňuje zaznamenať aktuálny stav virtuálneho stroja. Ak by v budúcnosti došlo k nejakej komplikácií, alebo strate dát, využitím snapshot-u dokážeme urobiť rýchlu obnovu dát k určitému bodu v čase.

4.1.2 Návrh

Na virtuálnom serveri je nainštalovaný operačný systém Ubuntu 17.04. Veľkosť operačnej pamäte je 2 GB, plus swap pamäť o veľkosti 1 GB. Celková veľkosť pevného disku (perzistentné úložisko) je 20 GB. Aby na serveri mohli bežať paralelne viaceré aplikácie, využili sme technológiu Docker.

Docker poskytuje ďalšiu vrstvu abstrakcie a automatizácie virtualizácie na úrovni operačného systému. Je to nástroj určený na uľahčenie vytvárania, nasadzovania a spúšťania aplikácií pomocou kontajnerov. Kontajnery umožňujú vývojárovi zabaliť aplikáciu so všetkými potrebnými časťami, ako sú napríklad knižnice a iné závislosti, do jedného balíka. Vďaka kontajneru je možné aplikáciu nasadiť na ľubovoľnom inom zariadení Linux bez ohľadu na to, aké nastavenia boli na zariadení, ktoré sa používalo na vytváranie aplikácie.

Na komunikáciu jednotlivých aplikácií so serverom sme sa rozhodli použiť architektonický štýl **REST**, pri ktorom sú jednotlivé správy vymieňané pomocou JSON a XML notácie. Dôvodom použitia práve REST-ových služieb je ich nezávislosť na platforme, čo umožňuje bezproblémové pripojenie rôznych druhov klientských aplikácií k serveru. Ďalej REST využíva jednoduché URI dopyty, čo umožňuje jednoduchšiu implementáciu klienta, ako napríklad využitie WSDL na popis webových služieb. V serverovej aplikácii sme definovali niekoľko rozhraní poskytujúcich služby pre klienta, ktorý ich volá podľa potreby.

Ak sa používateľ dopytuje po údajoch z databázy, prostredníctvom API jednoduchým volaním **GET** funkcie môže získať potrebné údaje. Taktiež, ak aplikácia bude chcieť ukladať zozbierané informácie na server, zavolaním funkcie **PUT** je možné tieto dáta perzistentne uložiť do databázy.

Predspracovanie (redukovanie) dát prichádzajúcich na server

Zariadenie na meranie dát umožňuje nastaviť pre rôzne snímače údajov rôzne frekvencie merania. Pre optický snímač a akcelerometer je to rozpätie 50 - 400Hz, pre teplomer 1 – 60s a pre ecg 128 – 512sps (samples per second). V rámci projektu sme sa rozhodli pre prvé tri uvedené snímače použiť najnižšiu možnú frekvenciu, s možnosťou neskoršieho skúmania vplyvu zvyšovania frekvencie na výsledok.

Údaje z optického senzora a akcelerometra prichádzajúce na server upravujeme z ľubovoľnej frekvencie na hodnoty prislúchajúce intervalu s dĺžkou 1s. Tie následne ukladáme do databázy vo formáte yyyy-MM-dd hh:mm:ss.

Spracovanie dát serverom

Pre určenie stresu je pre nás potrebné poznať kľudové hodnoty heart-rate-u v BPM (beats per minute) a okysličenia krvi v % daného používateľa. Oba tieto údaje získavame z dát optického senzora. Spracované charakteristiky daného používateľa ukladáme do databázy, a neskôr používame ako vstupné parametre výpočtov.

Odosielanie interpretovaných dát serverom

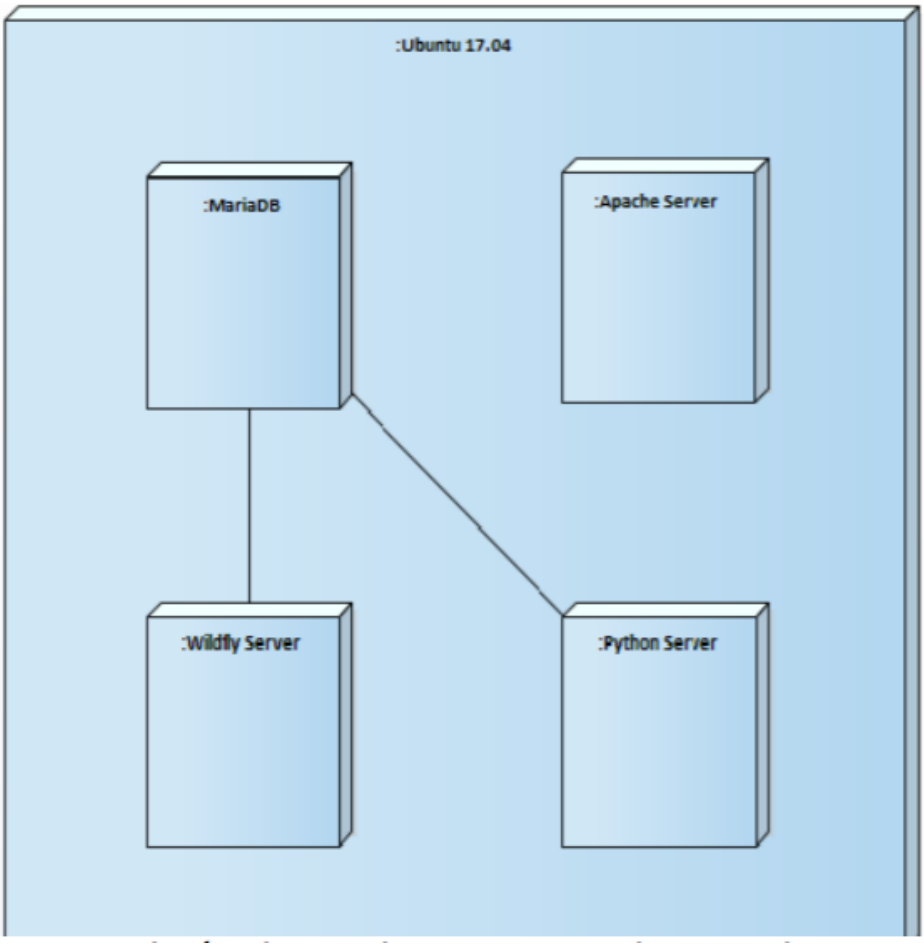
Server poskytuje pomocou restového rozhrania možnosť získať surové dáta - z optického senzora a akcelerometra priamy obsah tabuľky (x, y, z os resp RED, IR, GREEN dióda), alebo spracované/interpretované dáta. Pri akcelerometri sa jedná o priebežnú veľkosť zrýchlenia (ms^{-2}) vo zvolenom časovom rozpätí, vypočítanú ako veľkosť vektora medzi dvoma bodmi upravenú citlivosťou meracieho zariadenia. Pri optickom senzore sa dá získať okysličenie krvi meranej osoby v percentách, vypočítané ako pomer objemu okysličenej krvi (red dióda) ku neokysličenej krvi (infrared dióda) v meranej cievnej sústave upravený o priemerné hodnoty daného používateľa a citlivosť konkrétneho meracieho zariadenia. Údaj z teplomera nie je potrebné ďalej upravovať, nakoľko sa získava priamo v °C. Server ďalej poskytuje možnosť dopýtať sa na hodnoty heart-rate-u (v BPM) v zvolenom intervale, ktorý sa sleduje pomocou kolísania množstva absorbovaného svetla optického snímača priloženého na kožu, hodnoty stresu (v %, hodnota špecifická pre daného používateľa) v danom intervale, stanoveného pomocou sledovania srdcovej aktivity a odfiltrovaním predpokladanej fyzickej námahy vďaka údajom akcelerometra a okysličenia krvi (zostávajúci rozdiel je spôsobený psychickou námahou). Ďalšia restová služba odpovedá na dopyt potreby vyvolania odporúčania na zníženie stresu používateľa.

Vstupom je zvolený používateľ, časový bod, výstupom index typu odporúčania (odpočinok, príjem tekutín, motivačný citát, ...) stanovený na základe dennej doby a vývoja stresu v dobe pred zvoleným časovým bodom. Poslednou dátovou službou poskytovanou serverom je možnosť zavolať si jedným volaním "GetAll" funkcie naraz všetky údaje (SpO_2, teplota, heart-rate, stres, odporúčanie) zobrazované našou android aplikáciou v zvolenom intervale.

4.1.3 Implementácia

Na serveri bežia viaceré Docker kontajnery pomenované ako:

- **apache2** – slúži ako server, na ktorom beží tímová stránka,
- **django** – slúži ako python server, na ktorom beží webová aplikácia StressBusters,
- **mariadb** – slúži ako databázový server,
- **phpmyadmin** – slúži ako webové rozhranie na prístup k databáze,
- **wildfly** – slúži ako server komunikujúci s Android aplikáciou.



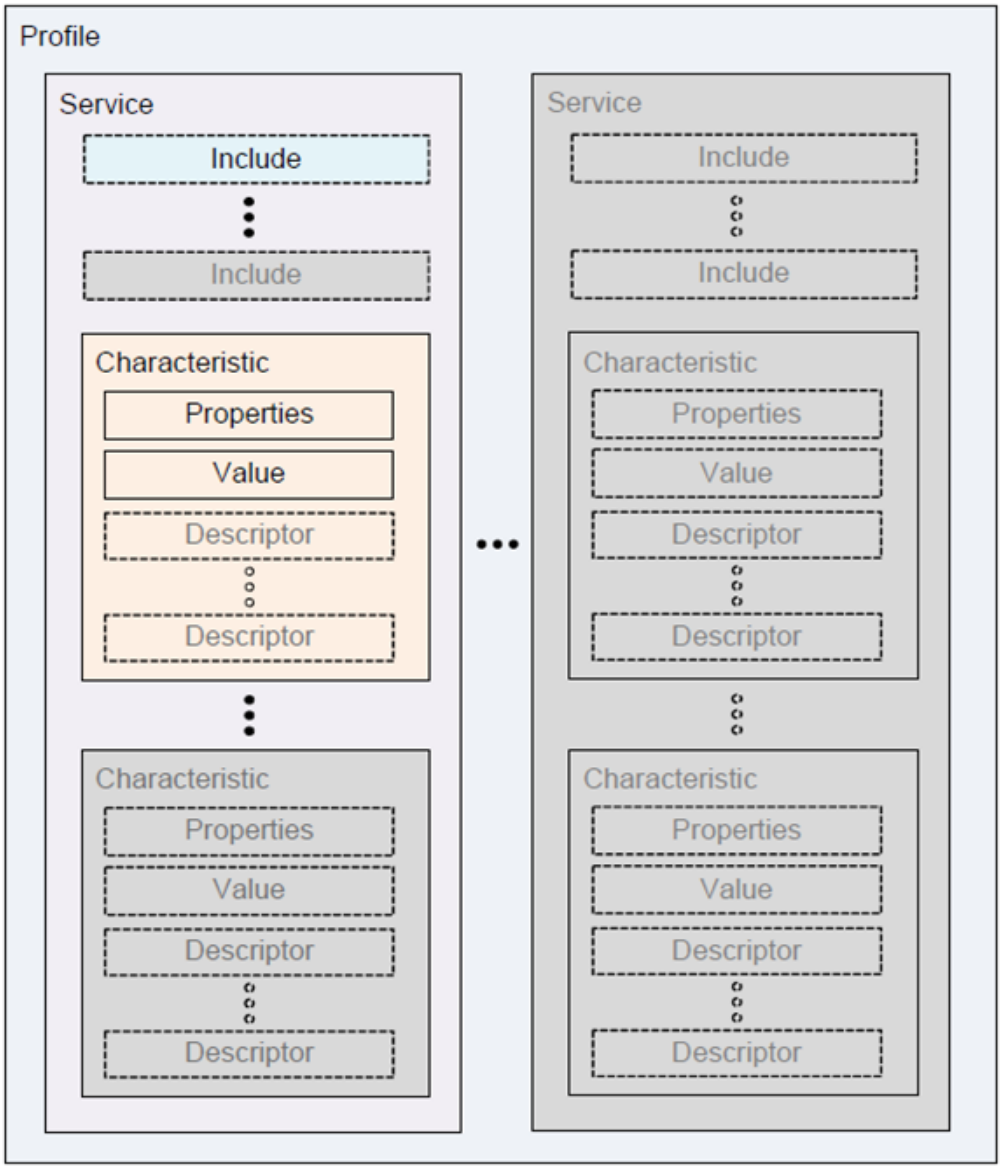
Obrázok 3: Diagram rozmiestnenia servera

4.2 Android aplikácia

4.2.1 Analýza

Bluetooth Low Energy

Bluetooth LE (Low Energy) je bezdrôtová komunikačná technológia, ktorá oproti klasickej Bluetooth technológii má nižšiu spotrebu energie. Je využívaná najmä v nositeľných (wearables) zariadeniach. Nie všetky zariadenia disponujúce s Bluetooth rozhraním podporujú túto technológiu.



Obrázok 4: GATT Profile Hierarchy [1]

Výmena údajov prebieha prostredníctvom **GATT** (Generic Attribute), ktoré sú založené na **ATT** (Attribute Protocol). "Používa hierarchickú dátovú štruktúru" [1]. Najvrchnejšia vrstva je profile, pod ňou sa nachádzajú services, ktoré ďalej obsahujú characteristics a tie ešte obsahujú descriptors. Services zoskupujú viaceré súvisiace atribúty ako napríklad **Device Information**, **Blood Pressure** alebo **Heart Rate**. "Charakteristiky sú údaje, ktoré zodpovedajú konkrétnemu internému stavu zariadenia alebo stavu prostredia, ktoré zariadenie môže odmerať použitím senzorov. Charakteristiky sa skladajú z viacerých častí. Majú typ, hodnotu, vlastnosti a nejaké povolenia."[2] Tie môžu predstavovať údaje ako Model Number, Blood Preasure Measurement alebo Heart Rate Max. V descriptors sa nachádzajú už len samotné metadáta. Všetky tieto údaje sú identifikovateľné prostredníctvom univerzálneho identifikátora UUID (Universal Unique Identifier).

Android ponúka svoje vlastné API na prácu s Bluetooth LE (Low Energy) zariadením. To poskytuje metódy na objavenie dostupných zariadení, pripojenie k nim, načítanie a zápis údajov.

Maxrefdes100 referenčná aplikácia

Na stránke produktu sa nachádzajú zdrojové kódy k referenčnej Android aplikácií. Jej úlohou je pripojenie mobilného zariadenia k doske cez Bluetooth LE (Low Energy) a zobrazenie získaných údajov. Pri spustení sa na prvej aktivite (obrazovke) zobrazí možnosť vyhľadania Bluetooth LE zariadení. Po pripojení k doske je zobrazená nová aktivita (obrazovka), ktorá graficky reprezentuje údaje získané zo zariadenia.

Komunikácia aplikácie so zariadením je zabezpečená prostredníctvom služby Bluetooth, ktorá má na starosti správu pripojenia s doskou a priebežne získavanie nameraných údajov. Keď nastane nová udalosť informuje o tom hlavnú aktivitu prostredníctvom broadcast správy a tá ju následne spracuje.

Aplikácia je schopná získať údaje z viacerých senzorov nachádzajúcich sa na zariadení. Sem patria:

- horný teplotný senzor,
- spodný teplotný senzor,
- akcelerometer,
- tlakový senzor,
- optický senzor,
- ECG.

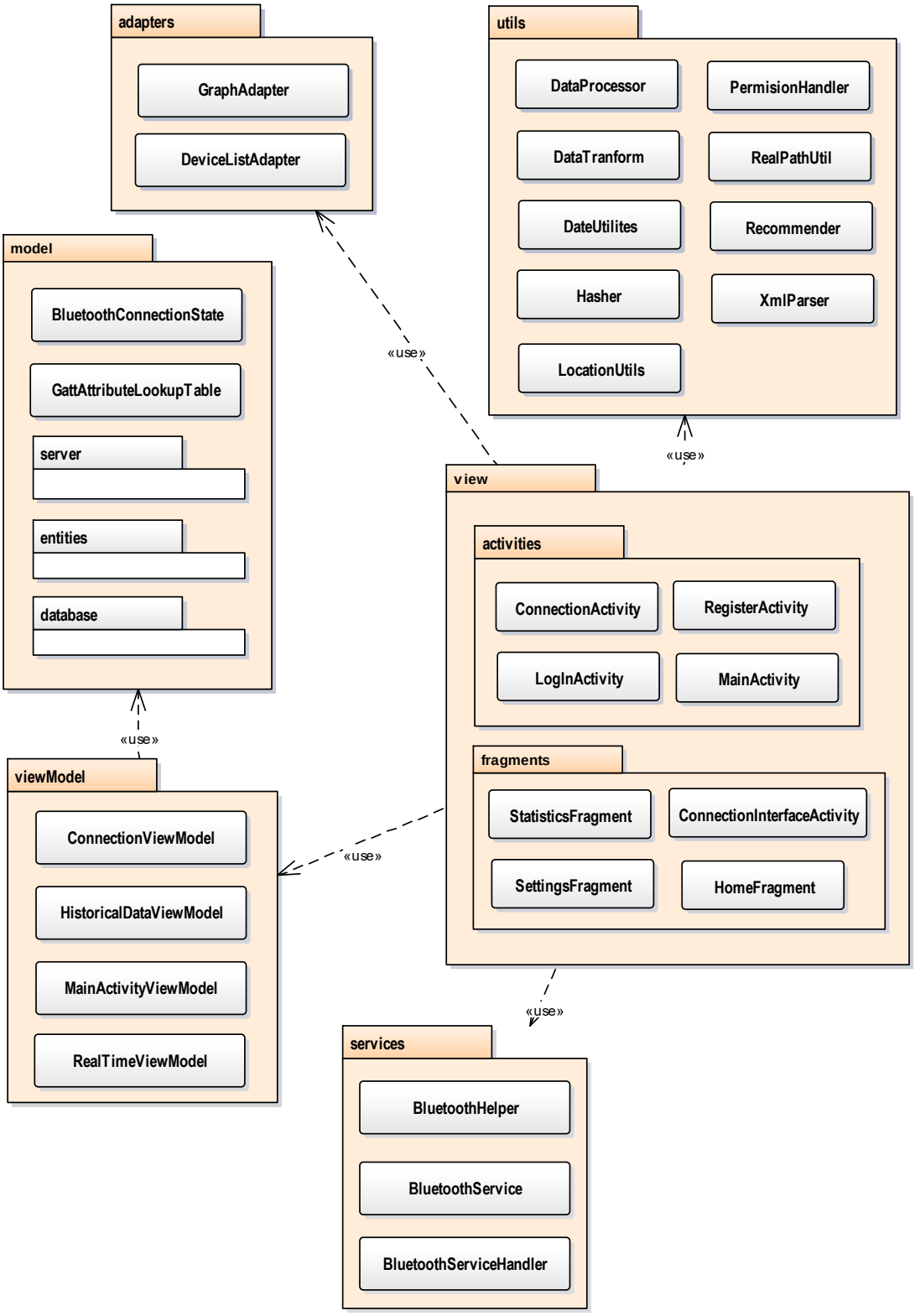
4.2.2 Návrh

Primárnym cieľom mobilnej aplikácie bude získavanie údajov zo servera a ich poskytnutie serveru. Ďalej bude schopná načítať spracované dáta zo servera, prípadne historické dáta prihláseného používateľa a vizualizovať ich. V neposlednom rade bude pri prekročení kritickej hranice stresu zobrazovať odporúčania pre zníženie hodnoty stresu.
Keďže sa v našom prípade jedná o citlivé osobné dáta našich používateľov, je potrebné zabezpečiť bezpečnú registráciu do systému a prihlasovanie používateľa do aplikácie.

4.2.3 Implementácia

V rámci aplikácie je zdrojový kód rozdelený do niekoľkých balíkov, ktorými sú:

- **model** – obsahuje dátové entity biometrických údajov ako aj rozhranie na ich ukladanie a získavanie zo servera alebo lokálnej databázy
- **services** – obsahuje triedy zabezpečujúcu komunikáciu so zariadením prostredníctvom Bluetooth spojenia vo forme služieb,
- **adapters** – nachádzajú sa v nej adaptéry zabezpečujúce buď grafickú reprezentáciu údajov alebo zobrazenie zoznamu zariadení,
- **utils** – obsahuje triedy slúžiace na spracovanie údajov ako napríklad parsovanie XML súborov, vyhodnotenie, kedy a aké odporučenie sa má používateľovi zobraziť, hashovanie hesla a iné.
- **view –** obsahuje aktivity a fragmenty slúžiace na prácu s používateľským rozhraním, ktoré je zobrazené používateľovi.
- **viewmodel –** obsahuje triedy určené na prepojenie grafického rozhrania view s dátami uloženými v modely. Sú tu uložené dáta určené na zobrazenie pre konkrétne grafické rozhranie. Sem patria štatistické dáta alebo aktuálne spracované dáta zo servera.



Obrázok 5: Diagram balíkov

4.2.3.1 Komunikácia

Komunikácia aplikácie so zariadením je zabezpečená prostredníctvom **Android Bluetooth Low Energy API**. Tá však vyžaduje viaceré oprávnenia ako android.permission.BLUETOOTH, android.permission.ACCESS_COARSE_LOCATION a iné. Sú deklarované v AndroidManifest.xml, ale pri novších verziach androidu je potrebné ich odsúhlasenie zo strany používateľa aj počas behu aplikácie, preto bola za týmto účelom vytvorená trieda PermissionHandler.

Na vyhľadávanie dostupných Bluetooth LE zariadení je využívaná metóda **discoverDevices** s časovačom na zastavenie, ktorá vracia všetky dostupné zariadenia. O pripojenie k vybranému zariadeniu sa stará služba **BluetoothService**, ktorá má na starosti správu pripojenia a získavanie údajov. O každej novej udalosti informuje prostredníctvom broadcast správ. Získané dáta odosiela triede **DataProcessor,** ktorá beží na samostatnom vlákne. Má na starosti neustále posielanie nameraných dát v dávkach v určitých časových intervaloch.

4.2.3.2 Vizualizácia

Na vizualizáciu údajov je využívaný **GraphView API**, ktorá umožňuje jednoduchú vizualizáciu údajov prostredníctvom svojich metód. Zobrazovanie údajov je možné v dvoch typoch. Prvým je zobrazenie surových načítaných dát zo senzora. V tomto prípade sú dáta zobrazené na čiarovom diagrame. Horizontálna os zobrazuje časový priebeh merania, vertikálna zobrazuje namerané dáta. Použitá API podporuje aj zobrazenie údajov z viacerých zdrojov v rámci jedného grafu, čo je využívané napríklad pri akcelerometri, ktorý pracuje s tromi údajmi, ktoré predstavujú jednotlivé osy x, y, z.

Druhým prípadom vizualizácie dát je zobrazenie už spracovaných dát. Spracované dáta sa zobrazujú na prstencových grafoch. Plný prstenec je zobrazený pri hodnote 100.

4.2.3.3 Dáta

Biometrické dáta sú ukladané v lokálnej **SQLite** databáze prostredníctvom triedy **LocalDatabase** využívajúcej knižnicu Room pre jednoduchšie rozhranie na komunikáciu s databázou. Pracuje s dátovými objektami, ktoré sú, buď ukladané do databázy alebo získavané z databázy. Súčasťou aplikácie je aj komunikácia so serverom. Využívaná je na zabezpečenie prihlásenia, registrácie používateľa, ale aj získavanie biometrických údajov konkrétneho používateľa. Celá tato implementácia je obsiahnutá v balíku Server. Keďže výmena údajov medzi mobilnou aplikáciou a serverom prebieha prostredníctvom správ vo formáte **XML**, bolo potrebné použitie špeciálnej triedy **retrofit2**. Tá ma na starosti transformáciu údajov medzi dátovými objektami a XML formátom správ.

4.2.3.4 Zníženie stresu

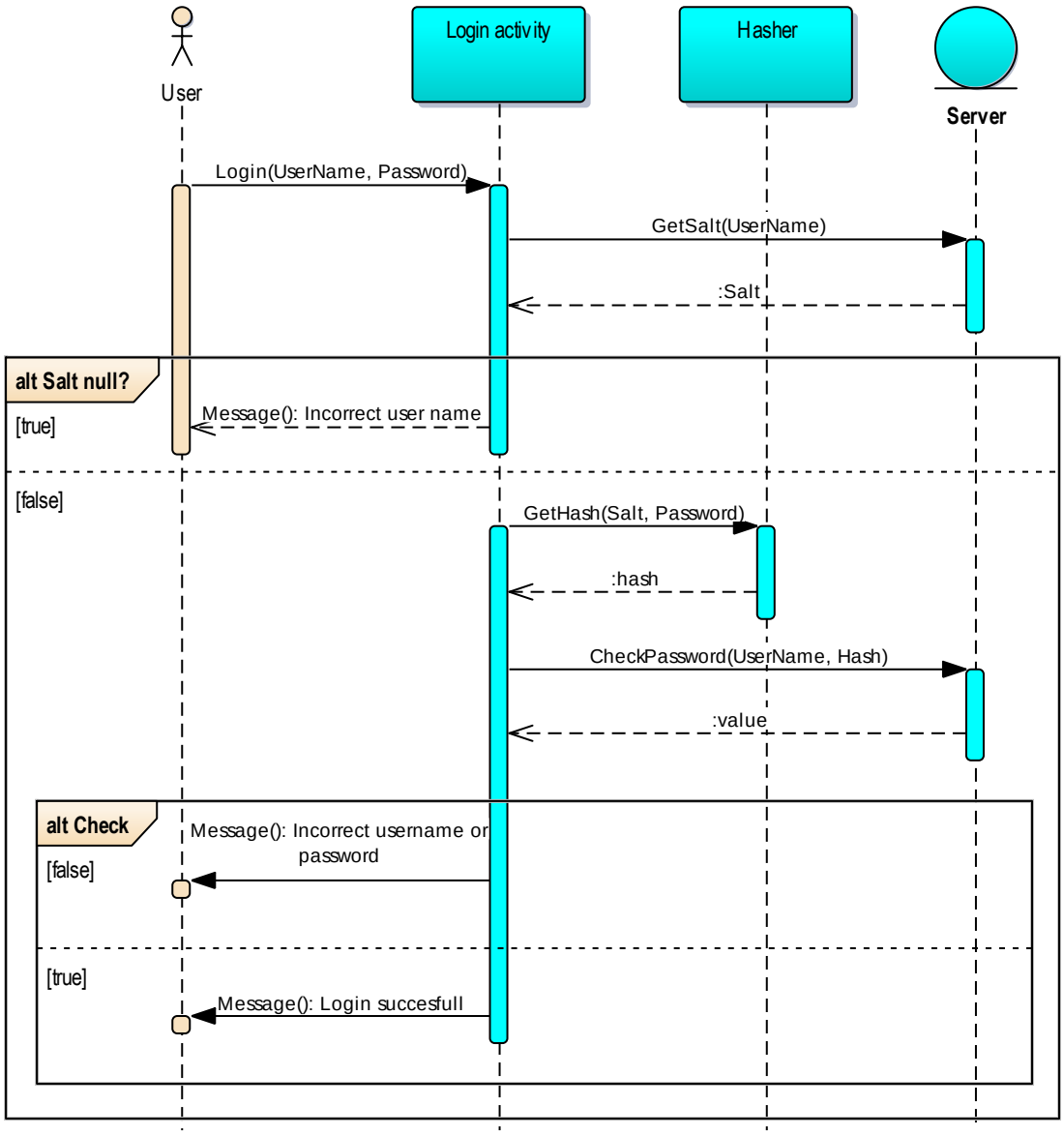
Veľmi významnou časťou vyvinutej aplikácie, na ktoré bol daný veľký dôraz, je odporúčanie pri zvýšenej hodnote stresu. Vyvinutá aplikácia umožňuje niekoľko typov odporúčaní. Sú to:

- motivačné citáty,
- krátke vtipné výroky,
- prehratie používateľom vybranej obľúbenej pesničky.

Všetky tieto formy zníženia stresu sú uložené lokálne v mobilnom zariadení, prípadne sú stiahnuté z internetu. Pri zvýšenej hodnote stresu server vyhodnotí aký typ je v danej situácii vhodný. Počas rozhodovania sa berie do úvahy úroveň stresu a taktiež aj informácia o poslednej poskytnutej alternatíve, aby nedošlo k duplicite pri po sebe idúcich odporúčaniach.

4.2.3.5 Prihlasovanie a registrácia

Pre použitie aplikácie je nutné byť zaregistrovaný a následne prihlásený z dôvodu, aby sme na serveri vedeli priradiť dáta konkrétnemu používateľovi. Keďže sa v minulosti plánovalo súčasne vyvíjať aj webovú aplikáciu, navrhli sme použiť rovnaký algoritmus, aký využíva Django. Tento proces nie je triviálny, preto jeho proces je názorne zobrazený na sekvenčnom diagrame na obrázku č.6. Kvôli tomuto procesu bola implementovaná aj trieda Hasher, ktorá dokáže zahashovať heslo pomocou rovnakého algoritmu ako Django. Pri šifrovaní sa používa hashovací algoritmus SHA-256. Vývoj webovej aplikácie bol síce pozastavený, ale pre možnosť ďalšieho vývoja v budúcnosti bol tento spôsob prihlasovania používaný aj naďalej.



Obrázok 6: Prihlásenie používateľa

4.2.4 Testovanie

Testovanie aplikácie bolo vykonávané už počas vývoja. Malé celky boli pred úspešným pridaním do hlavnej vetvy dôsledne otestované členmi tímu.

Prvé veľké testovanie väčšieho celku bolo vykonané počas veľkého 8-hodinového testu načítavania údajov zo senzora a ukladanie do lokálnej databázy. Na tieto dáta boli následne aplikované metódy na určenie stresu testovaného jedinca. Počas celého testovania aplikácia plnila svoju funkciu bez komplikácii.

Druhým veľkým testovaním prešla aplikácia pri odovzdaní zadávateľovi tohto projektu. V tejto fáze sa už testovalo aj posielanie dát v reálnom čase cez pripojenie do internetu. Počas tohto testovania došlo k občasnému prerušeniu posielania dát. Všetky tieto nedostatky boli zaznamenané a v ďalších fázach vývoja opravené.

4.3 Webová aplikácia

Táto kapitola obsahuje opis webovej aplikácie. Ako však bolo spomenuté v úvode, v letnom semestri sa vývoj webovej aplikácie zamedzil. Preto táto časť obsahuje súhrn webovej aplikácie len zo zimného semestra a ponúka možnosť jej rozvoja v rámci iných projektov.

4.3.1 Analýza

Kvôli veľkému množstvu nameraných hodnôt a ich analýze prichádza otázka kde spracovávať údaje. Prenášanie veľkého množstva dát zo servera ku klientovi, ktoré by sa spracovávali na strane klienta je rovnako diskutabilné ako by sme spracovávali žiadosti od veľkého počtu klientov na strane servera, ktorý musí ukladať nové údaje od mobilných aplikácií zároveň.

4.3.2 Návrh

Webová aplikácia slúži na výmenu údajov medzi používateľom a serverom. Mala by byť schopná registrácie nových používateľov, prihlasovania a odhlasovania ale hlavne vizualizácie a analyzovania nameraných hodnôt používateľa krátkodobo aj dlhodobo. Riešením na otázku v analýze je navzorkovanie dát tak aby došlo k minimálnej strate informácií, ktoré sa z nich dajú získať ale zároveň aby sa dalo čo najviac zredukovať množstvo dát, ktoré je potrebné preniesť.

4.3.3 Implementácia

Na implementáciu sme sa rozhodli použiť framework **Django** na rýchli vývoj webových aplikácií. Vývoj v Djangu väčšinou prebieha v jazyku **Python** ale dizajn mu dodávajú **HTML** a **CSS**, pričom sa vo veľkej miere využíva aj framework **Bootstrap**. Po problémoch s nasádzaním na server sme sa dohodli využiť softvér **Docker**. Je to nástroj na vytváranie kontajnerov (podobné virtuálnym obrazom - image), ktoré ak človek rozchodí u seba na počítači tak pomocou pár príkazov môže vytvorený projekt testovať alebo nasadzovať na serveri alebo na inom počítači.

Inicializácia webovej aplikácie prebiehalo založením nového projektu v Djangu. Pre lepšiu prácu na vývoji aplikácie a pri vytváraní dockerovského imagu, bolo potrebné nainštalovať izolované pythonovské prostredie **virtualenv** a manažéra balíkov **pip**, pomocou ktorého dokážeme vytvoriť zoznam všetkých potrebných závislostí, ktoré sa nachádzajú vo virtuálnom prostredí. Postupnosť krokov, ktoré sa musia vykonať na vytvorenie image-u sa popisuje v súbore **Dockerfile**. Súbor **docker-compose.yml** definuje ako sa majú rôzne kontajnery správať, napríklad v našom prípade komunikácia databázy s webovou aplikáciou.

Nová funkčnosť sa vytvára príkazom **django startapp** <meno>, ktorá vytvorí kostru pre danú funkčnosť. Vizuálne prvky k danej funkčnosti (napr. formátovaný formulár pre login) sa definujú v priečinku **templates**, s tým, že sa tam vytvorí priečinok s rovnakým menom ako je meno funkčnosti.

Webová aplikácia okrem prihlasovania podporuje aj registráciu používateľa. Aby sa žiadala aj emailová adresa od používateľa pri registrácii, bolo nutné rozšíriť základnú triedu **UserCreationForm** o email a následne ešte upraviť ukladanie údajov do databázy pri úspešnej registrácii.

Komunikácia so serverom prebieha pomocou **REST** api. Na získanie dát zo strany servera sa posiela požiadavka pomocou balíčka **requests**. Ako odpoveď server posiela dáta v xml formáte, ktoré sú následne rozparsované. Získané dáta sa vizualizujú pomocou **Chartjs API**.

4.3.4 Testovanie

Testovanie prebiehalo pomocou dockeru. Po vytvorení kontajneru a dokončení implementácie boli súbory nahrané na git a zodpovedný člen tímu za code review vykonal testovanie. Testovanie zahŕňalo stiahnutie súborov, vyskúšanie vytvorenia a rozbehnutia kontajneru na vlastnom počítači. Následne kontroloval vytvorené zdrojové kódy a nové pridané funkcie.

ZDROJE

[1] – *Bluetooth: GATT overview* [online]. [cit. 10.11.2017]. Dostupné na: https://www.bluetooth.com/specifications/gatt/generic-attributes-overview

[2] – Wooley, M. *Bluetooth blog: A Developers Guide To Bluetooth* [online]. 10. Aug. 2016 Dostupné na: http://blog.bluetooth.com/a-developers-guide-to-bluetooth

Inštalačná príručka

Inštalácia mobilnej aplikácie

Mobilnú aplikáciu StressMonitor je možné nainštalovať pomocou priloženej aplikácie **StressMonitor.apk** na stránke projektu. Aplikáciu je potrebné stiahnuť do mobilného zariadenia s Android-om a v ňom ju nainštalovať.

Inštalácia servera

Na inštaláciu servera wildfy, mysql ako aj apache2 je pripravený docker súbor (**`docker-compose.yml`**), ktorý je potrebné aj so zdrojovými súbormi uploadnuť na server. V tomto súbore `docker-compose.yml` je potrebné upraviť globálnu premennú *MYSQL_ROOT_PASSWORD*, ktorej hodnotu je taktiež potrebné zmeniť v konfigurácií wildfly.

Inštaláciu servera je možné spustiť príkazom **`docker-compose build`**, čím sa spustí build jednotlivých serverov a stiahnu sa potrebné repozitáre z oficiálnej stránky *https://hub.docker.com/*.

Po úspešnom dokončení predchádzajúceho procesu je potrebné spustiť príkaz **`docker-compose up -d`**. Týmto príkazom sa spustia jednotlivé servery. Prepínač -d spôsobí, že jednotlivé procesy sa spustia na pozadí. Pre overenie a výpis súčasných bežiaci procesov je možné zadať príkaz **`docker ps`**.

Používateľská príručka

1 Prihlásenie/registrácia

Spustením aplikácie sa zobrazí prihlasovacie okno, ktoré slúži na prihlásenie používateľa do aplikácie. Obrazovka obsahuje dve polia na vyplnenie používateľského mena a hesla. Používateľ môže zaškrtnúť možnosť „**Remember me**“ (v preklade zapamätaj si ma), na uloženie prihlasovacích údajov. Po vyplnení týchto údajov sa môže používateľ prihlásiť pomocou tlačidla „**LOG IN**“. Na prihlásenie je potrebné pripojenie k internetu. Ak používateľ nie je pripojený k internetu alebo zadal zlé prihlasovacie údaje, je upozornený systémom.

Obrázok 7: Prihlásenie

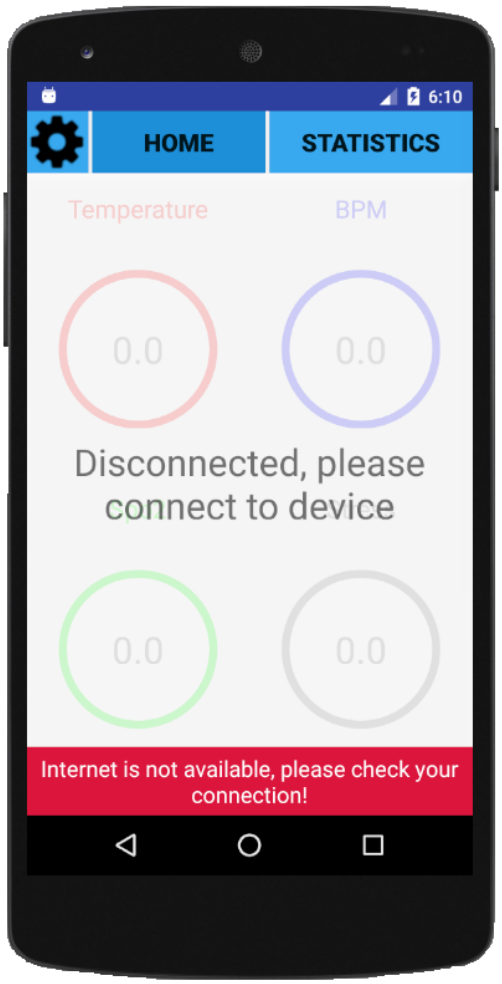
Obrázok 8: Registrácia

Ak používateľ ešte nepoužíval našu aplikáciu a nemá založené konto na našom serveri, môže využiť možnosť registrácie pomocou tlačidla „**SIGN UP**“. Na tejto obrazovke sa nachádzajú polia na vyplnenie používateľského mena (**User name**), krstného mena (**First name**), priezviska (**Last name**), E-mailu, hesla (**Password**) a potvrdenie hesla (**Confirm password**). Heslo a potvrdenie hesla sa musia zhodovať, inak registrácia nebude úspešná. Takisto si používateľ nemôže zvoliť také isté používateľské meno, aké v systéme už existuje. Po správnom vyplnení polí sa používateľ môže zaregistrovať stlačením tlačidla „**SIGN UP**“.

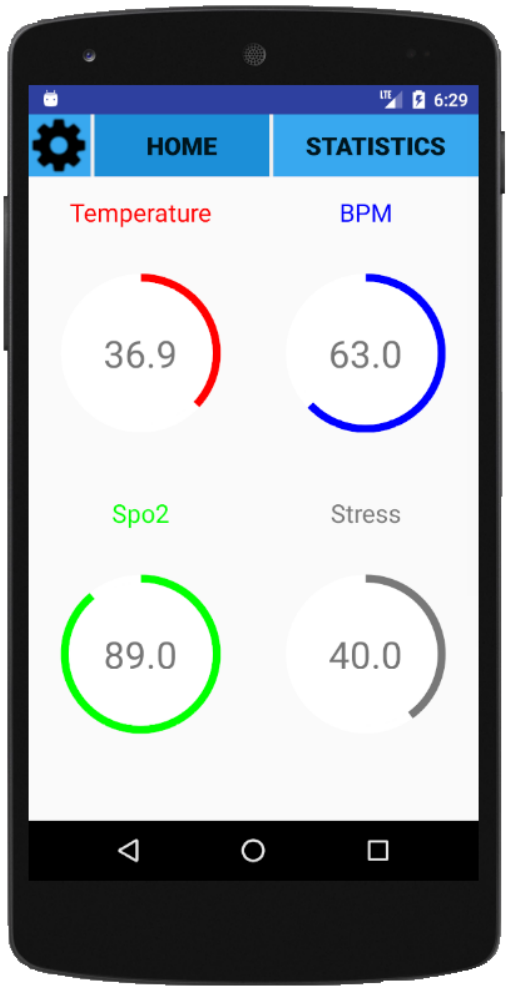
2 Hlavné obrazovky aplikácie

Po úspešnom prihlásení sa používateľ dostane na obrazovku domov – **HOME**, ktorá je zobrazená na obrázku č. 3 a obrázku č. 4. Ak používateľ nie je pripojený k žiadnemu zariadeniu na meranie, zobrazí sa mu hláška: *„Disconected, please connect to device“*. V tomto prípade je potrebné sa pripojiť k zariadeniu v nastaveniach aplikácie, ako je opísané v nasledujúcej kapitole.

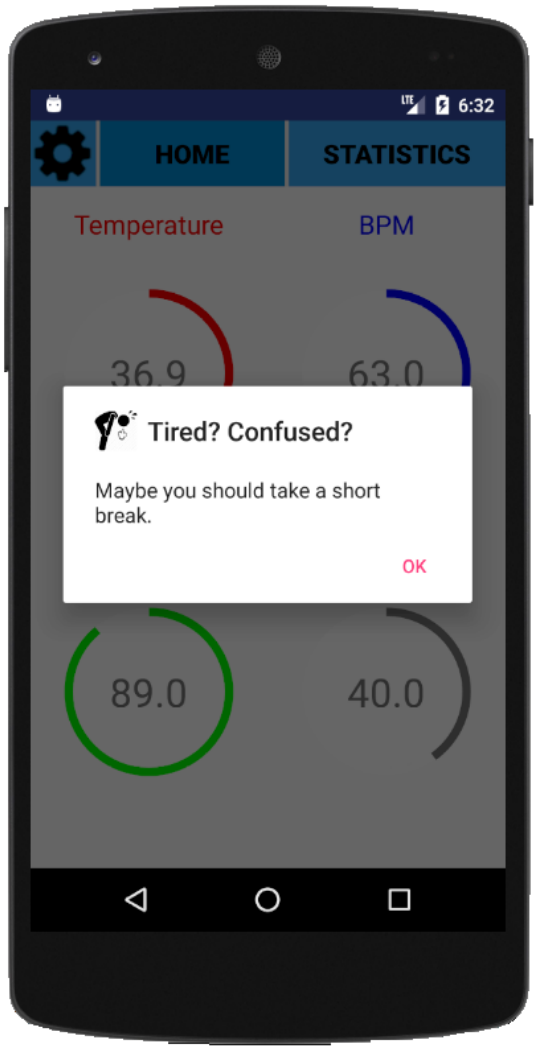
Ak je používateľ pripojený k zariadeniu, môže si zobraziť údaje zo zariadenia v reálnom čase. Táto obrazovka obsahuje 4 grafy, ktoré znázorňujú aktuálne hodnoty namerané z pripojeného zariadenia. Tieto hodnoty sú telesná teplota, heart-rate v BPM SPO2 a stres.



Obrázok 3: Obrazovka Home bez pripojenia



Obrázok 4: Obrazovka Home

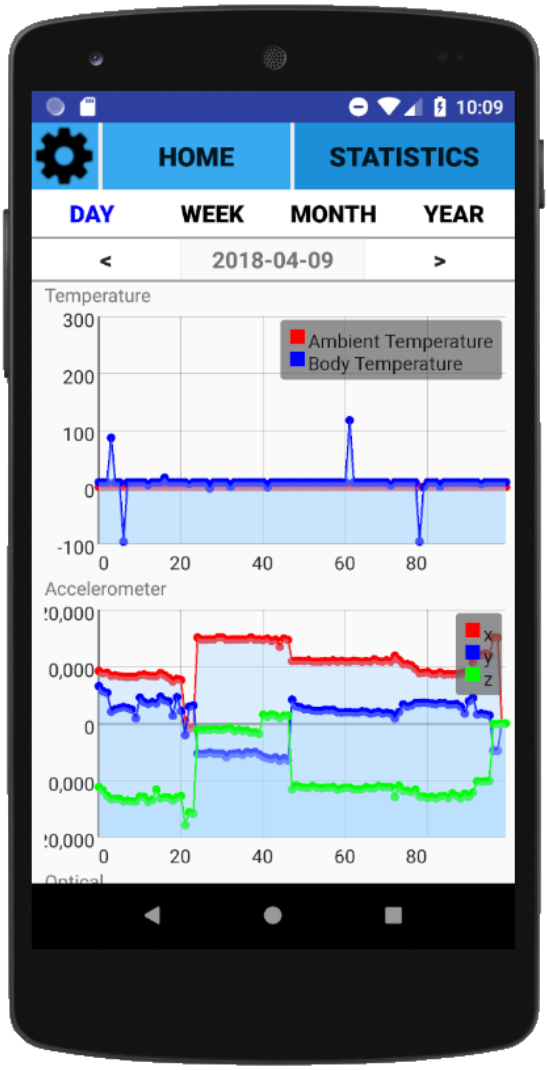


Obrázok 5: Zobrazenie odporúčania

alebo **YEAR** (rok). Časový interval môže posúvať pomocou príslušných šípok alebo stlačením dátumu, čím sa zobrazí kalendár. V ňom si môže vybrať konkrétny dátum. Tým sa následne zobrazia údaje z vybraného dátumu.

Pri zvýšenom strese sa používateľovi zobrazí motivačný citát, krátky vtipný výrok alebo sa prehrá jeho obľúbená pesnička. Príklad je zobrazeného výroku je na obrázku č. 5.

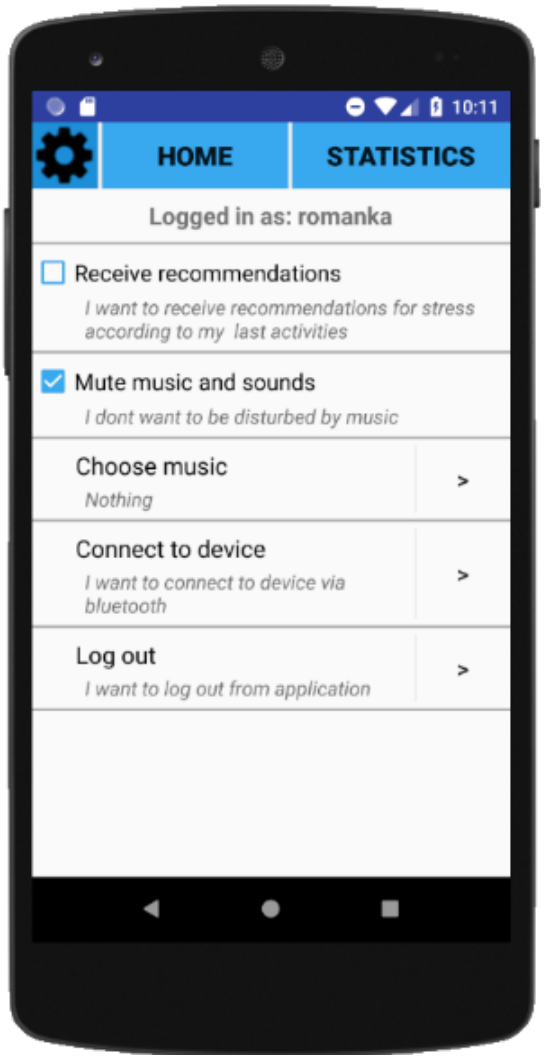
Druhou obrazovkou je obrazovka **STATISTICS**. Tu si môže používateľ pozrieť svoje namerané historické dáta. Zobrazenie je prehľadné v troch grafoch, ktoré reprezentujú postupne teplotu tela a okolia, údaje z akcelerometra a z optického senzora. Používateľ má možnosť si upraviť časové rozpätie, v akom sú zobrazené dáta a to stlačením príslušného tlačidla **DAY** (deň), **WEEK** (tyždeň), **MONTH** (mesiac)



Obrázok 6: Obrazovka Statistics

3 Nastavenia

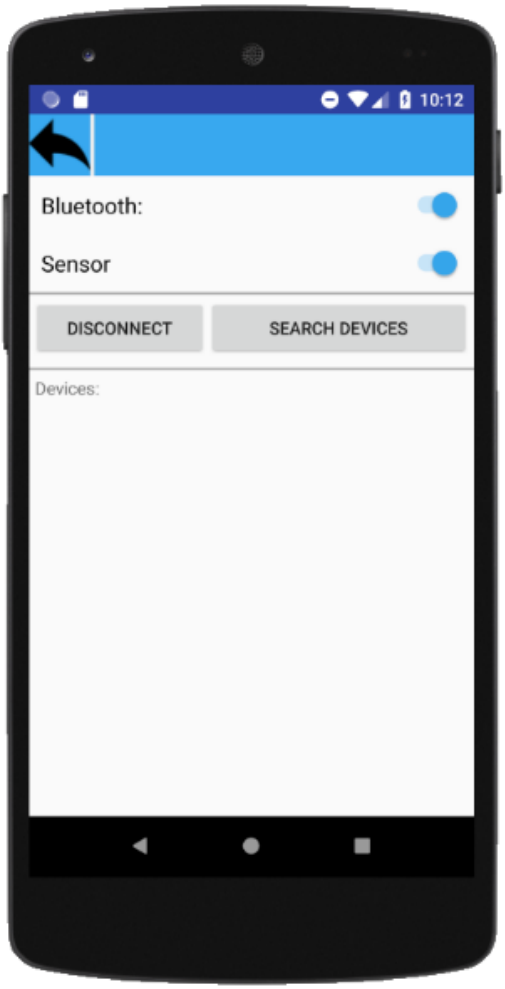
Kliknutím na ikonu skrutky sa používateľ dostane do nastavení aplikácie. Prvým možným nastavením je spustenie prijímania odporúčaní pri zvýšenom strese. Zaškrnutím tejto možnosti sa používateľovi pri zvýšenom strese zobrazí napríklad motivačný citát, odporúčanie, aby sa napil alebo sa spustí jeho obľúbená pesnička. Tieto odporúčania sú doprevádzané zvukovými efektami, ktoré môže stĺmiť ďalším nastavením – **Mute music and sound** (stlm hudbu a zvuky).



Obrázok 7: Obrazovka Settings

Svoju obľúbenú pesničku, ktorá sa môže spustiť pri zvýšenom strese, sa nastaví možnosťou **Choose music** (výber hudby). Výberom tejto možnosti sa používateľ dostane do súborového systému svojho telefónu, z ktorého si môže vybrať jednu z dostupných pesničiek. Názov vybranej pesničky sa následne zobrazí pod názvom tejto možnosti. Ak používateľ nemá vybranú žiadnu skladbu, namiesto názvu je zobrazený text **Nothing**.

Jednou z prvých veci, ktorú musí používateľ spraviť po úspešnom prihlásení, je pripojenie sa k zariadeniu, z ktorého bude prijímať dáta. To môže vykonať pomocou možnosti v nastaveniach **Connect to device**. Výberom tejto možnosti sa presmeruje na ďalsiu obrazovku ako je zobrazené na obrázku č. 6. Tu má používateľ možnosť zapnúť (prípadne vypnúť) bluetooth. Ak má zapnutý bluetooth, môže vyhľadať dostupné zariadenia pomocou stlačenia tlačidla „**Search devices**“ (vyhľadaj zariadenia). Stlačením sú následne vyhľadané zariadenia, ktoré majú taktiež zapnutý bluetooth. Výberom jedného zariadenia sa aplikácia k tomuto zariadeniu pripojí. Ak sa používateľ chce odpojiť od zariadenia, môže tak vykonať tlačidlom „**Disconnect**“. Na danej obrazovke je aj možnosť zapnutia senzoru, po pripojení sa cez bluetooth k senzoru sa pomocou tohto nastavenia zapne optický senzor a senzor na meranie pohybu, na zariadení. Tieto senzory je nevyhnutné zapnúť pre korektné fungovanie aplikácie.



Obrázok 8: Obrazovka Connect to device

Pod hlavným menu sa nachádza text „**Logged in as**“, za ktorým sa nachádza používateľské meno práve prihláseného používateľa. Ak sa chce odhlásiť, môže tak vykonať pomocou možnosti „**Log out**“.