Slovenská technická univerzita

Fakulta informatiky a informačných technológií

Ilkovičova 2, 842 16 Bratislava 4

Tímový projekt

Breyslet

Inžinierske dielo

Akademický rok:	2018/2019
Vedúci tímu:	Ing. Kunštár Vladimír
Členovia tímu (tím č. 3):	Bc. Andráš Radovan Bc. Chlepková Michaela Bc. Gábriš Daniel Bc. Lančarič Viktor Bc. Lauko Juraj Bc. Petráš Ján Bc. Uhnáková Veronika Bc. Valach Alexander

Zoznam skratiek

ADC - Analog-to-Digital Converter

AES - Advanced Encryption Standard

ALU - Arithmetic Logic Unit

API - Application Programming Interface

EEPROM - Electrically Erasable Programmable Read-Only Memory

DAC - Digital-to-Analog Converter

DES - Data Encryption Standard

GPIO - General Purpose Input/Output

GPR - General Purpose Register

GPRS - General Packet Radio Service

GPS - Global Positioning System

GSM - Global System for Mobile Communications

I2C - Inter-Integrated Circuit

MCU - Microcontroller Unit

MIPS - Milion Instructions per Second

PDI - Program and Debug Interface

PLL - Phased-Locked Loop

PWM - Pulse-Width Modulation

RTC - Real-Time Clock

SPI - Serial Peripheral Interface

SPM - Store Program Memory

SRAM - Static Random Access Memory

TWI - Two-Wire Interface

UART - Universal Asynchronous Receiver-Transmitter

UI - User Interface

Obsah

1. Úvod

Tento dokument predstavuje dokumentáciu k projektu Breyslet. Ide o vnorený, stavovo-informačný systém monitorovania zdravia osôb za pomoci náramku, ktorý zaznamenáva zdravotné údaje a vyhodnocuje potenciálne riziká pre nositeľa zariadenia.

Dokument obsahuje analýzu danej problematiky. Ďalšou časťou je špecifikácia požiadaviek na systém ako aj návrh riešenia. V neposlednom rade dokument opisuje spôsob implementácie jednotlivých častí a prototypu. Dokument obsahuje aj záznam o testovaní prototypu.

Tento projekt vznikol ako zadanie z predmetu Tímový projekt v akademickom roku 2018/2019.

2. Globálne ciele projektu

Naším cieľom je vytvorenie systému, ktorý bude monitorovať zdravie osôb a vyhodnocovať potenciálne riziká. Systém je určený hlavne pre starších ľudí, ktorí často žijú sami a nemá sa o nich kto postarať.

Systém pozostáva z viacerých častí. Prvou časťou je vytvorenie zariadenia, ktoré bude merať dáta o aktuálnom zdravotnom stave nositeľa náramku. Tieto dáta sú napríklad tep, okysličenie krvi, teplota a podobne. Okrem toho bude zariadenie vedieť privolať pomoc v prípade zaznamenaného pádu a bude vybavený tlačidlom pre privolanie pomoci. Ďalšou časťou je vytvorenie aplikácií, pomocou ktorých môžu rodinní príslušníci kontrolovať zdravotný stav svojich blízkych aj na diaľku. Okrem toho treťou časťou systému bude dohľadové centrum, ktoré bude dostávať upozornenia v prípade, že zariadenie zaznamenalo pád alebo nositeľ stlačil tlačidlo na privolanie pomoci. Centrum toto upozornenie preverí a zabezpečí pomoc pre nositeľa náramku. Ďalšími časťami systému sú server, databáza a časti potrebné na komunikáciu.

2.1. Globálne ciele projektu pre ZS

Našim cieľom v zimnom semestri akademického roka 2018/2019 je analyzovanie a navrhnutie hlavných častí systému. Medzi tieto časti patrí návrh a implementácia dátového modelu a návrh dosky plošného spoja. Ďalej sem patrí návrh a realizácia základných častí firmvéru (komunikácia s perifériami, USART, I2C). Návrh a implementovanie serverovej časti. Taktiež navrhnutie používateľských rozhraní pre mobilné aplikácie a web.

2.2. Globálne ciele projektu pre LS

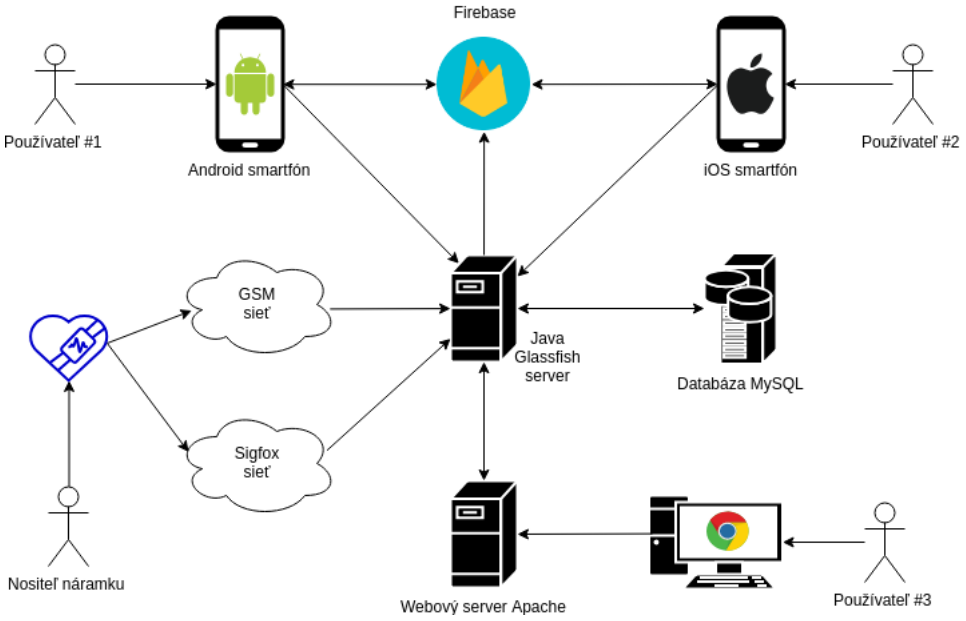
Po zimnom semestri bolo dobre rozpracovaných niekoľko funkčných celkov hlavne firmvér, práca s hardvérom a práca na serveri. Podarilo sa nám naplniť plán, ktorý sme si stanovili na začiatku semestra pre prvú časť práce na projekte. V práci sme pokračovali aj počas skúškového obdobia, nakoľko je projekt veľmi rozsiahly.

Na začiatku letného semestra sme si stanovili plán práce a ciele, ktoré sme chceli dodržať. Plánom pre letný semester je dokončenie firmvérovej časti projektu, ako aj návrh dosiek plošných spojov. Samozrejme cieľom je aj funkčná serverová časť a prípadné drobné úpravy databázy, ktorá je však po zimnom semestri už hotová. Vytvorenie mobilných aplikácií pre Android a IOS, ako aj vytvorenie webovej aplikácie je tiež na pláne v tomto semestri.

3. Celkový pohľad na systém

3.1. Architektúra systému

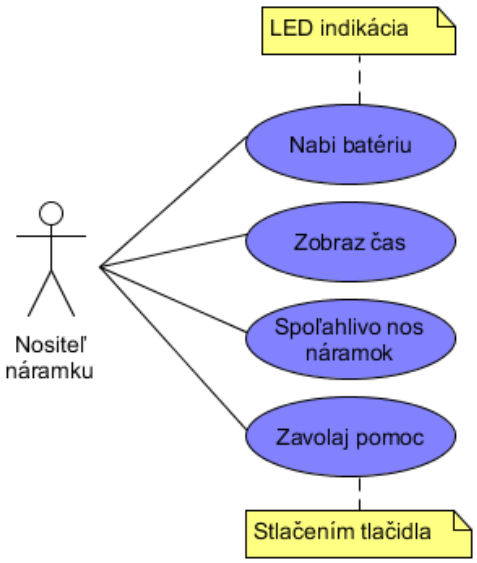
Architektúra vnoreného, stavovo-informačného systému.



Obr. 1: Architektúra systému

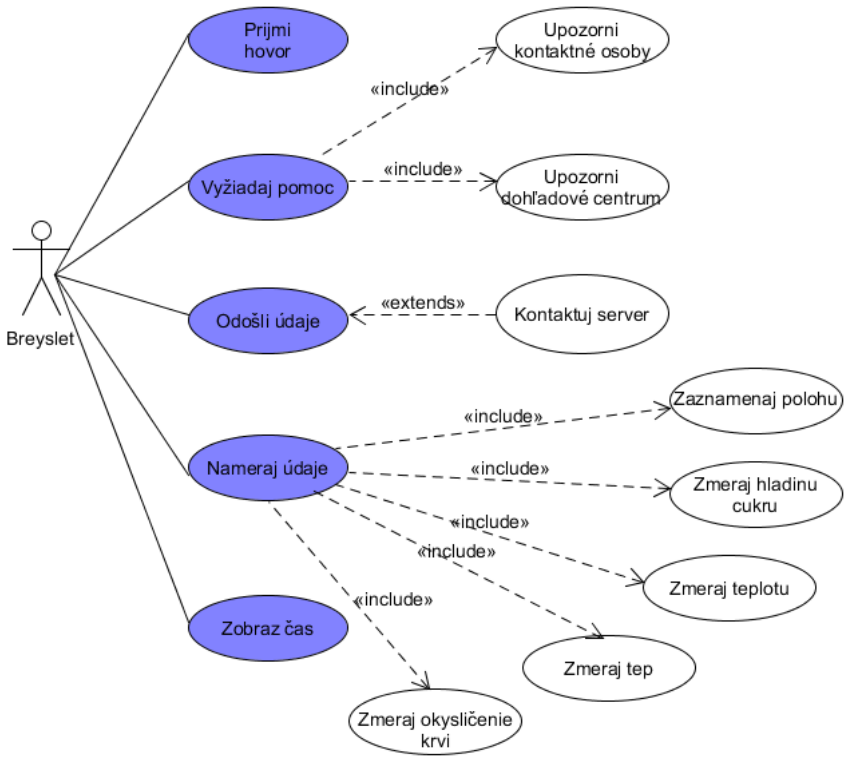
3.2. Prípady použitia

V tejto časti sa identifikujú prípady použitia celého navrhovaného systému.



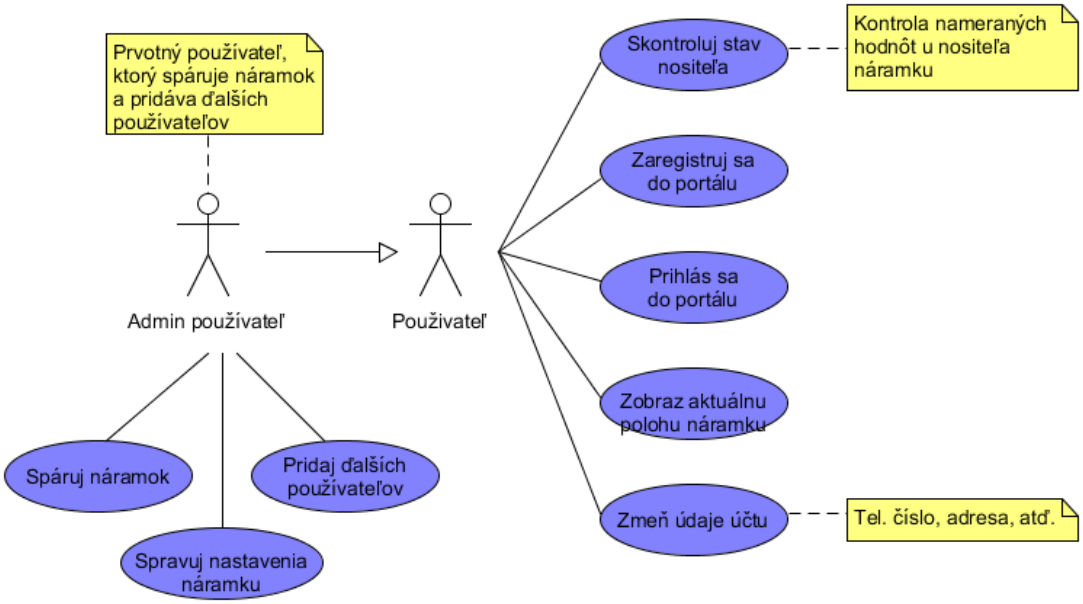
Obr. 2: Diagram prípadov použitia - nositeľ náramku

Diagram na obrázku (Obr. 2) znázorňuje funkcionality, ktoré s náramkom môže vykonávať jeho nositeľ. Nositeľ má priradený prípad použitia spoľahlivo nosiť náramok kvôli tomu, že jednak musí byť náramok pohodlný na nosenie, ale zároveň musí byť pevný a odolný.



Obr. 3: Diagram prípadov použitia – Breyslet

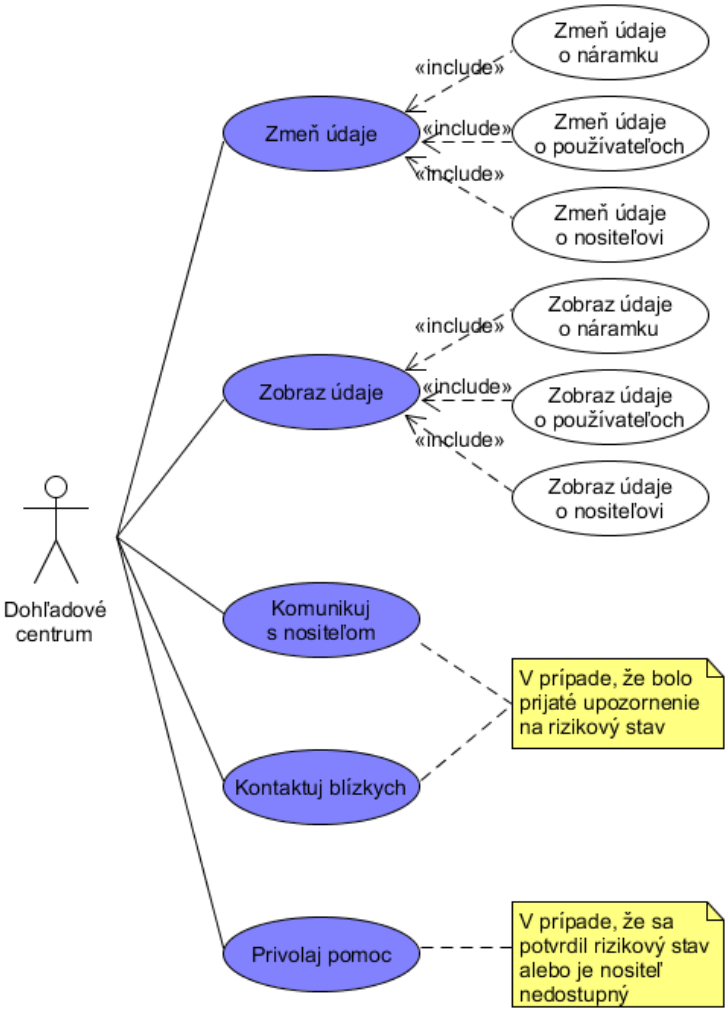
Obr. 3 znázorňuje funkcie samotného náramku, teda čo všetko bude vykonávať bez ohľadu na nositeľa. Hlavnou funkciou je meranie hodnôt životných funkcií, ktoré sa pravidelne odosielajú na server pre spracovanie. V prípade rizikových hodnôt okamžite vyžiada pomoc, teda zašle upozornenie kontaktným osobám a dohľadovému centru. Kontaktné osoby môžu aj na základe tohto upozornenia kontaktovať nositeľa náramku, prípadne privolať zdravotnú pomoc.



Obr. 4: Diagram prípadov použitia – používateľ

Tretí diagram (Obr. 4) znázorňuje vytvorenie dvoch úrovní používateľských účtov. Prvá úroveň je správcovský používateľ, ktorý ako jediný má možnosť spárovať náramok a pridať ďalších používateľov pre sledovanie tohto náramku. Druhú úroveň predstavuje obyčajný používateľ, ktorý má možnosť sledovať namerané životné funkcie osoby nosiacej náramok a zobraziť polohu náramku.

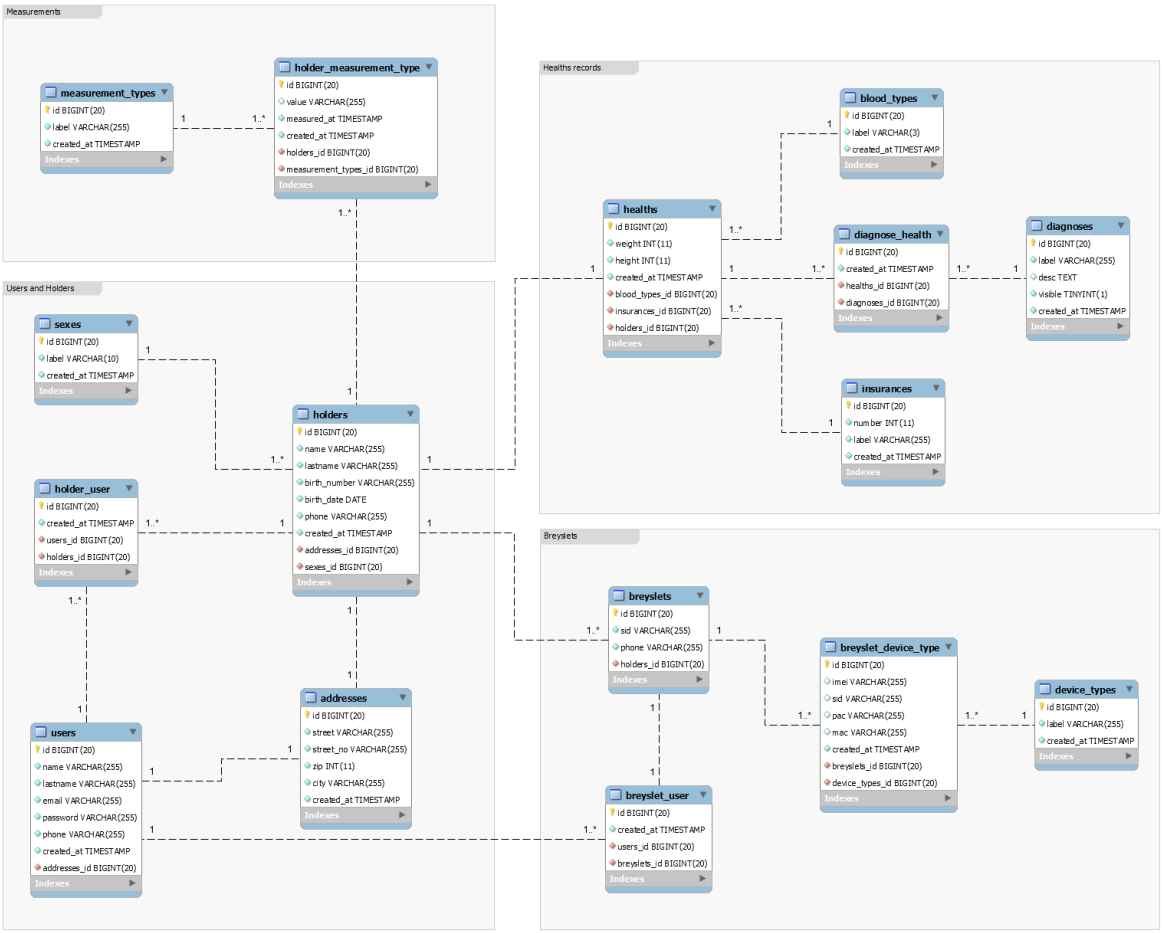
Posledný diagram (Obr. 5) popisuje možnosti a funkcie dohľadového centra, ktoré sleduje stav nositeľov a tiež majú možnosť meniť údaje jednak náramku, nositeľa, ale aj používateľa. V prípade rizikového stavu majú povinnosť kontaktovať nositeľa, prípadne kontaktné osoby a privolať pomoc.



Obr. 5: Diagram prípadov použitia - dohľadové centrum

3.3. Dátový model

Jedným zo základných prvkov počítačového systému je databáza. Tá slúži na zhromažďovanie údajov zo zariadení a je potrebná pre správnu funkčnosť systému. Nasledujúci obrázok 6 zobrazuje fyzický dátový model.

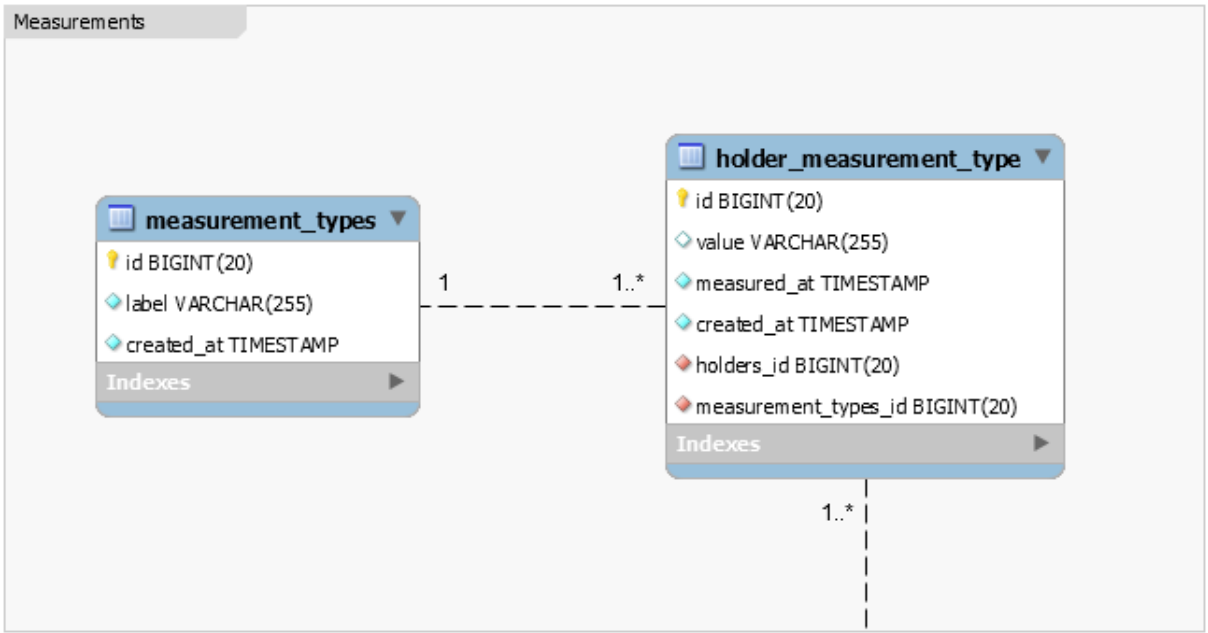


Obr. 6: Dátový model

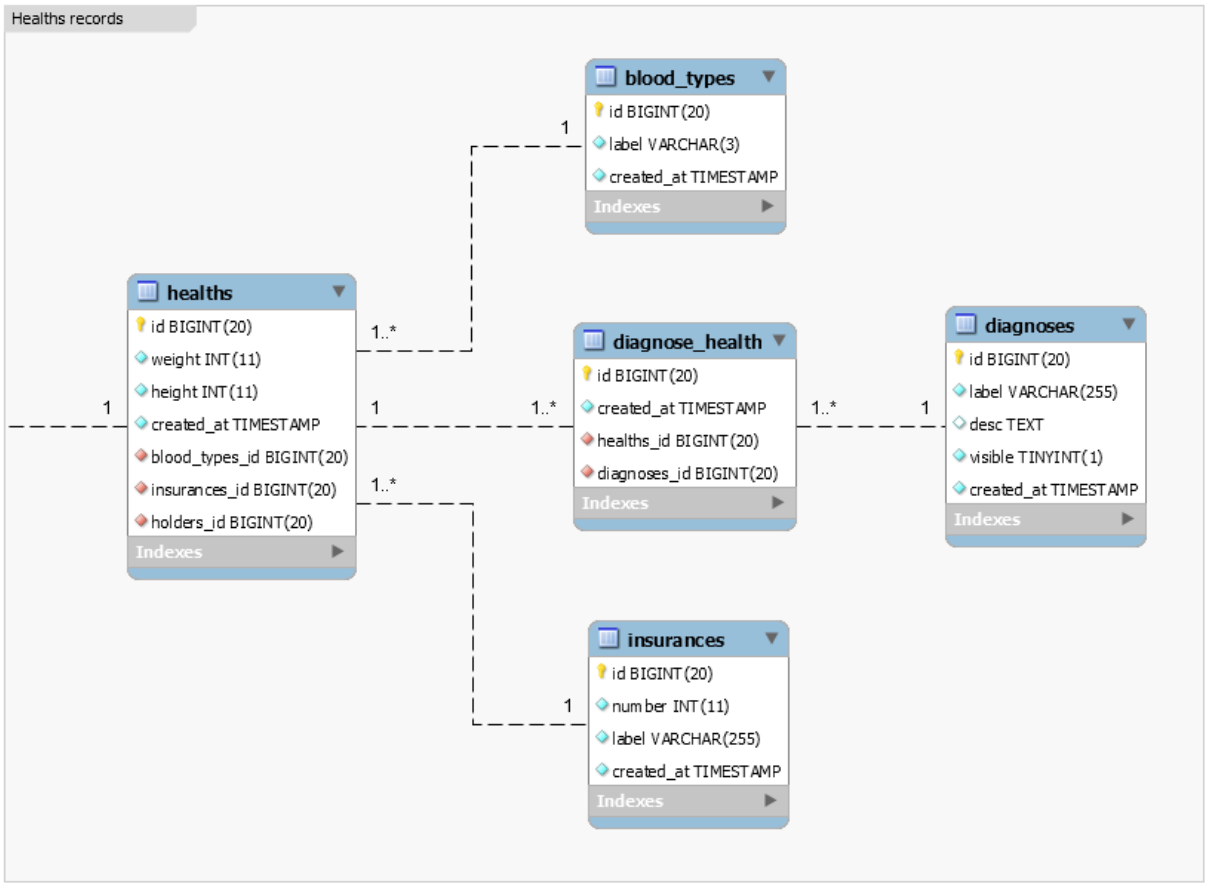
Najdôležitejšími entitami v databáze sú náramky „breyslets“, používatelia „users“ a držitelia „holders“. Držitelia sú tí, ktorí náramky nosia a používatelia tí, ktorí ich spravujú. Jednotlivé náramky sa skladajú z viacerých komponentov o ktorých si potrebujeme uchovávať dôležité informácie, ako napríklad výrobné čísla. O držiteľovi náramku si budeme uchovávať základné zdravotné informácie, ako sú typ krvnej skupiny, diagnózy, váha, výška a taktiež akú má poisťovňu. Jednotlivé merania z náramku sa budú ukladať do tabuľky meraní, kde bude uvedené o aký typ merania sa jedná. Používateľ bude naviazaný na držiteľa

a na náramok. V prípade že bude existovať vzťah medzi používateľom a náramkom to znamená, že je používateľ administrátorom daného náramku.

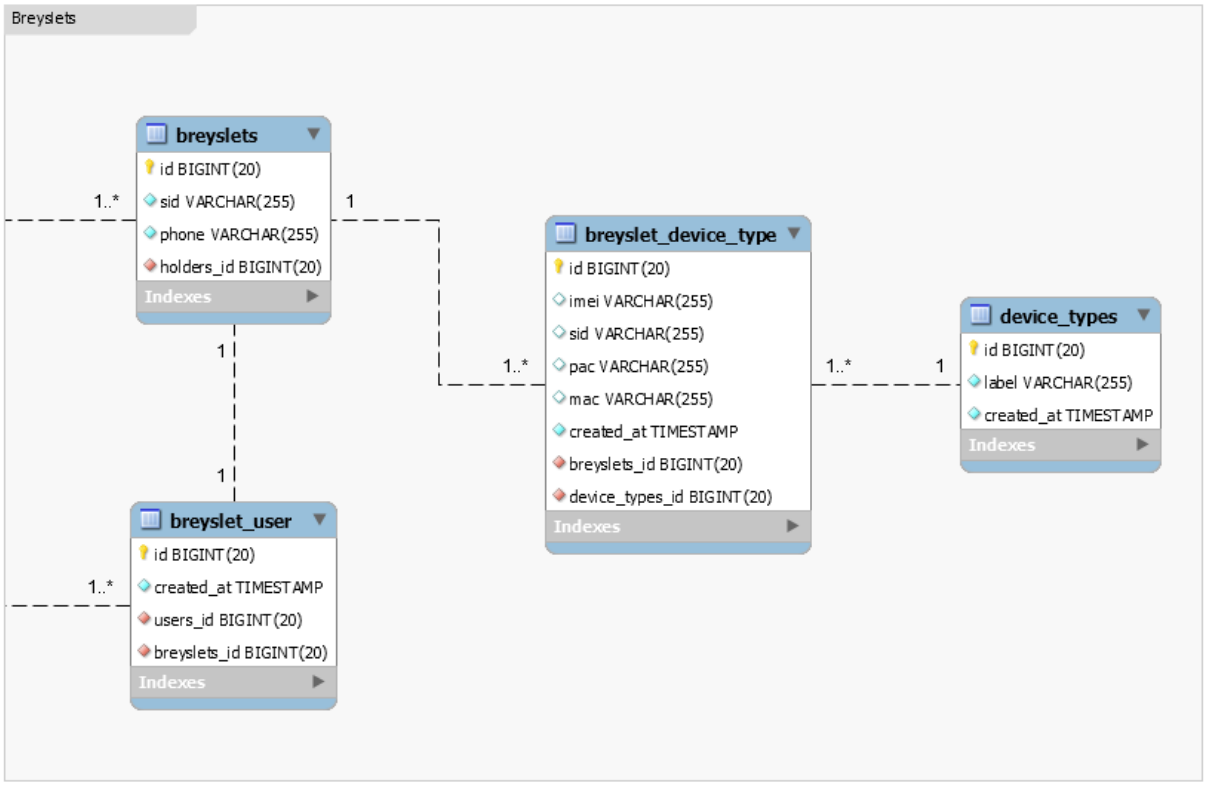
Nižšie na obrázkoch sú zobrazené jednotlivé logické bloky fyzického dátového modelu pre lepšiu čitateľnosť.



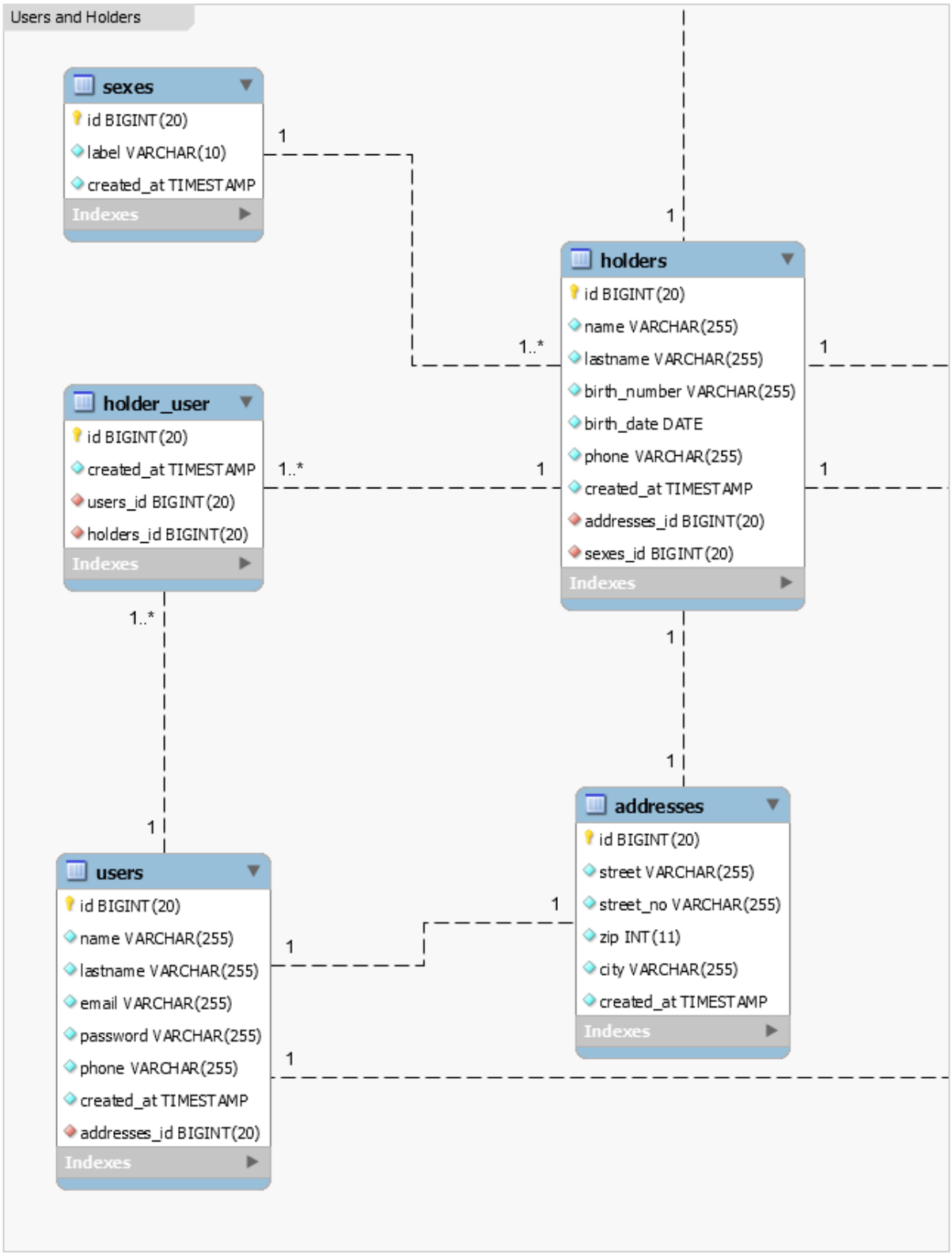
Obr. 7: Tabuľky meraní



Obr. 8: Tabuľky zdravotných záznamov



Obr. 9: Tabuľky breysletov

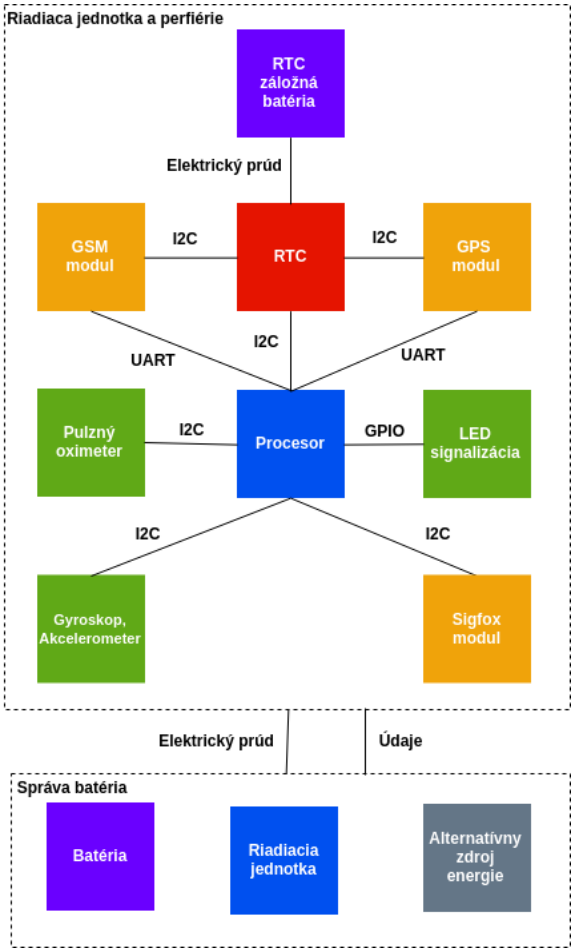


Obr. 10: Tabuľky používateľov a držiteľov

4. Moduly

4.1. Hardvér

Samotný vnorený systém bude pozostávať z niekoľkých hardvérových komponentov znázornených na obrázku (Obr. 11).



Obr. 11: Riadiaca jednotka

4.1.1. ATxmega128A4

Patrí do rodiny procesorov Atmel AVR XMEGA A4. Je to nízkoenergetický a vysokovýkonný mikrokontrolér založený na AVR vylepšenej architektúre. XMEGA A4 dosahuje priepustnosť 1 MIPS za každý MHz, čo poskytuje dizajnérovi optimalizovať spotrebu energie pomocou pracovnej frekvencie. Procesor kombinuje bohatú inštrukčnú sadu s 32 GPU registrami, ktoré sú priamo pripojené k aritmeticko-logickej jednotke. Toto umožňuje dva nezávislé prístupy do pamäte v jednej inštrukcii, počas jedného hodinového cyklu.

AVR Xmega A4 zariadenia poskytujú nasledujúce funkcionality:

- Zabudovaná programovateľná flash pamäť s funkciou „read-while-write“
- Interná EEPROM a SRAM
- 4-kanálový DMA kontrolér
- 8-kanálový systém udalostí a viacúrovňový kontrolér prerušení.
- 34 všeobecných vstupno/výstupných liniek
- 16 bitový RTC (Real time counter)
- 16 bitový časovač/počítadlá s porovnávacími módmi a PWM
- 5 UART rozhraní
- 2 TWI (Two-wire serial interface)
- 2 SPI rozhrania
- AES a DES šifrovacie nástroje
- 12-kanálový, 12 bitový ADC, s programovateľným zosilnením
- 2-kanálový 12 bitový DAC
- 2 analógové komparátory s window módom
- Programovateľný watch dog timer so samostatným interným oscilátorom
- Presný interný oscilátor s PLL s deličom.
- PDI rozhranie

XMEGA A4 zariadenia majú nasledovných päť softvérovo voliteľných úsporných režimov:

- Idle – CPU a nonvolatilné pamäťe sú zastavené, avšak všetky periférie, kontrolér prerušení, event system a DMA kontrolér sú aktívne. Akékoľvek prerušenie zobudí zariadenie.
- Power-down – Všetky hodinové signály sú zastavené, vrátane RTC. Toto umožňuje prácovať iba asynchrónnym modulom. Prerušenia, ktoré dokážu zobudiť MCU sú prerušenia cez TWI a asynchrónne porty.
- Power-save - Rovnaký mód ako power-down, s tým rozdielom, že ak je povolené RTC, tak ostane fungovať. Zariadenie môže byť tým pádom zobudené aj pretečením RTC alebo Compare match prerušením.
- Standby – Tento mód je identický s power-down módom s výnimkou toho, že povolené zdroje systémových hodín ostanú aktívne. Tým sa zabezpečí znížený čas potrebný pre prebudenie.

- Extended standby – Je rovnaký ako power-save mód. Výnimku v tomto móde tvoria povolené zdroje systémových hodín. CPU a periférne hodiny sú zastavené. Tiež znižuje čas prebudenia.

Pre fungovanie týchto režimov procesor využíva tzv. Power Reduction Register.

4.1.2. GSM

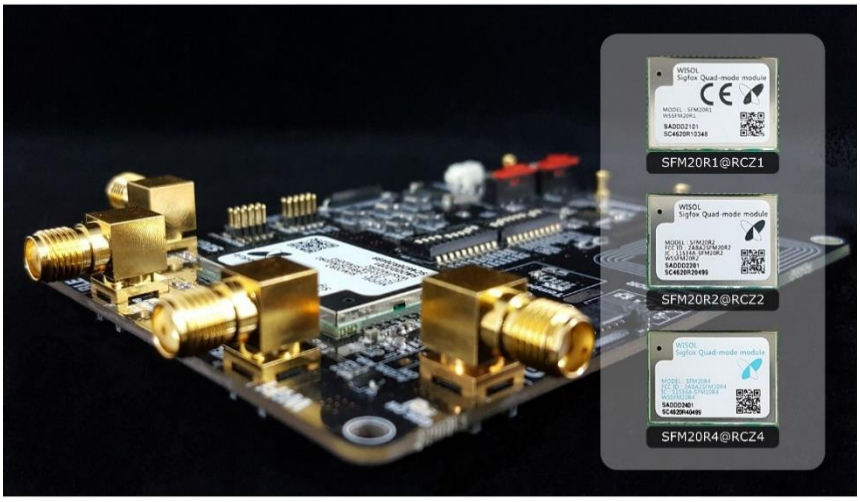
GSM je celosvetovo rozšírená mobilná sieť s výborným pokrytím, ktoré je pre náš účel veľmi dôležité. Taktiež spoľahlivosť tejto siete je na vysokej úrovni. Nielen tieto dva faktory zavážili pri výbere toho, akú technológiu použijeme na komunikáciu medzi náramkom a serverom. Bežné odosielanie dát bude prebiehať prostredníctvom siete Sigfox, pretože spotreba energie pre posielanie dát touto sieťou je nižšia ako cez sieť GSM. Avšak GSM sieť budeme používať v prípade potreby odoslať väčšie množstvo dát alebo v prípade nejakého incidentu. Dáta budeme odosielať prostredníctvom GPRS, čo je technológia prepínania paketov v GSM sieti.

Pre potreby nášho zariadenia sme zvolili čip M66 od spoločnosti Quectel, pretože oproti konkurencii prevažuje v oblasti spotreby, veľkosti a hmotnosti. Pre porovnanie sme uviedli aj čip Fibocom G510 Q50-00.

Tabuľka 1: Porovnanie parametrov čipov

Parametre/Typ	M66	G510 Q50-00
Napájanie	3,3 ~ 4,6 V	3,3 ~ 4,5 V
GSM	850/900/1800/1900 MHz	850/900/1800/1900 MHz
Rozhranie	UART	UART
Režimy	IDLE, SLEEP, DATA	IDLE, RTC, SLEEP, POWER OFF, GPRS data
Odber pri komunikácii	250 mA	350 mA
Odber v režime SLEEP	13 mA	1,3 mA
Odber v režime IDLE	13 mA	22 mA
Rozmery	15.8 x 17.7 x 2.3 mm	20.2 x 22.2 x 2.5 mm
Hmotnosť	1,3 g	2.5 g

4.1.3. Sigfox - vývojový kit, EVBSFM20R1



Obr. 12: Sigfox – vývojový kit

4.1.3.1. Analýza

Pre uľahčenie prototypovania používame vývojový kit EVBSFM20R1. Vývojový kit disponuje hneď niekoľkými čipmi. Nachádza sa v ňom Wi-Fi, Bluetooth Low Energy, GPS, akcelerometer a už spomínaný Sigfox. Vďaka vývojovému kitu vieme s jednotlivými súčasťami komunikovať a ovládať ich.

Dôvodom, prečo sme sa rozhodli práve pre Sigfox je, že je to IoT technológia, ktorá má veľké pokrytie siete. V Európe pracuje na frekvencií 868MHz. Prenos dát v sieti Sigfox je obmedzený na uplink 12B a maximum 140 správ a downlink 8B a 4 správy za deň. Nie sú to veľké hodnoty, no pre nás sú postačujúce.

4.1.3.2. Návrh

Otestujeme na vývojovom kite komunikáciu s čipom. Keď už budeme vedieť plne spravovať a ovládať všetky potrebné súčasti, tak zo samotného vývojového kitu použijeme len samotný čip SFM20R1, ktorý obsahuje Sigfox, Wi-Fi, Bluetooth Low Energy, GPS a akcelerometer. Pre tento čip budeme následne potrebovať mať navrhnutú vlastnú základovú dosku, do ktorej čip osadíme a kde s ním budeme ďalej vedieť komunikovať a pracovať.

4.1.3.3. Implementácia

Pre začatie práce s čipom je potrebné naštudovanie dokumentácie a dlhé hodiny testovania a skúšania komunikácie pomocou AT príkazov. Podarilo sa nám nadviazať

komunikáciu cez I2C rozhranie, cez ktoré vieme posielať AT príkazy čipu Sigfox a ten ich následne odošle do siete Sigfox. Vzhľadom na to, že maximálny počet správ, ktoré môžeme cez sieť Sigfox poslať je obmedzený, tak sme si museli vypočítali časové intervaly, v ktorých môžeme tieto správy posielať. Ďalším postupom bude osadenie čipu do nami navrhnutej dosky.

4.1.4. Signalizačný modulu pre dohľadové centrum

V projekte budeme potrebovať aj mikrokontrolér, ktorý by mal GPIO vstupy a výstupy a podporuje USB zbernicu. Z produktového katalógu mikrokontrolérov od firmy Microchip sme vybrali nasledujúce dva AT90USB647 a ATmega8U2. V nasledujúcej tabuľke sú uvedené parametre oboch mikrokontrolérov.

Tabuľka 2: Porovnanie mikrokontrolérov

	ATmega8U2	**AT90USB647**
Program Memory Type	Flash	Flash
Program Memory Size (KB)	8	64
CPU Speed (MIPS/DMIPS)	16	16
SRAM Bytes	512	4
Data EEPROM/HEF (bytes)	512	2048
Digital Communication Peripherals	1-UART, 2-SPI,	1-UART, 2-SPI, 1-I2C
Capture/Compare/PWM Peripherals	1 Input Capture, 1 CCP, 5PWM	1 Input Capture, 1 CCP, 10PWM
Timers	1 x 8-bit, 1 x 16-bit	
Number of Comparators	1	1
Number of USB Modules	1, Full Speed	1, None
Temperature Range (C)	-40 to 85	-40 to 85
Operating Voltage Range (V)	2.7 to 5.5	2.7 to 5.5
Pin Count	32	64

Z údajov vieme určiť, že pre naše potreby je postačujúci mikrokontrolér ATmega8U2. Keďže potrebujeme lacný mikrokontrolér s USB a 10 GPIO vstupmi/výstupmi.

4.1.5. Odhadovaná spotreba energie

Tabuľka 3: Spotreba energie

Modul	Čip	Spotreba energie [mA]	Poznámky
GSM modul	Quectel M66	1.3	standby
Gyroskop	MPU-6050	3.6	
Akcelerometer	MMA8451Q	0.165	
Procesor	ATxmega128A4	200	
Pulzný oximeter	MAX30105	1.1	
Sigfox	nRF52832	5.4	maximum RX
Displej	1.54inch e-paper	10	

Odhadovaná spotreba je 221.565 mA.

4.2. Firmvér

4.2.1. Impelementácia

Firmvér vnoreného systému Breyslet je písaný v jazyku C. Manifestom pri tvorbe firmvéru nám bola priložená príručka **XMEGA-C-MANUAL.pdf**. V programe využívame nasledovné štandardné knižnice:

1. **AVR IO** <avr/io.h> - definície jednotlivých konštánt pre prácu s AVR procesormi.
2. **UTIL DELAY** <util/delay.h> - funkcie na správu s časom
3. **STD IO** <stdio.h> - práca s reťazcami znakov

Okrem toho využívame aj vlastné knižnice:

1. TWI MASTER (twi_master.h) - slúži na ovládanie I2C zbernice (riadenie komunikácie s teplomerom a systémovými hodinami).
2. USART (usart.h) - slúži na odoslanie výstupu z mikrokontroléra na USART.
3. Knižnica na prácu so systémovými hodinami (rtc.h).
4. Knižnica na ovládanie pulzného oximetra (oximeter.h) – obvod používame momentálne ako merač teploty, ale kvôli ďalšiemu vývoju projektu nesie knižnica meno podľa hlavnej činnosti obvodu.

5. Knižnica na prácu so zdrojom hodín (clock.h).

6. Knižnica pre prácu s modulom Sigfox (sigfox.h)

7. Knižnica pre prácu s GSM modulom (gsm.h)

8. Knižnica na ovládanie senzora na meranie telesnej teploty (temperature.h)

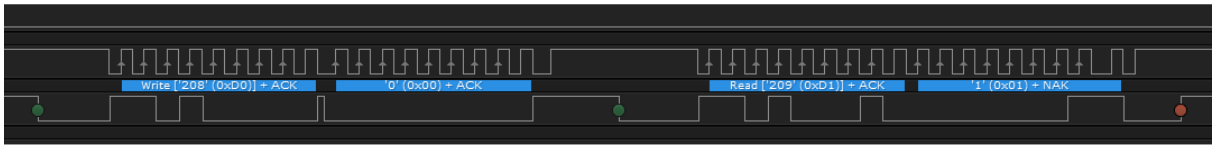
4.2.2. Testovanie

V nasledujúcej časti uvádzame protokol pre testovanie.

4.2.2.1. Test systémových hodín

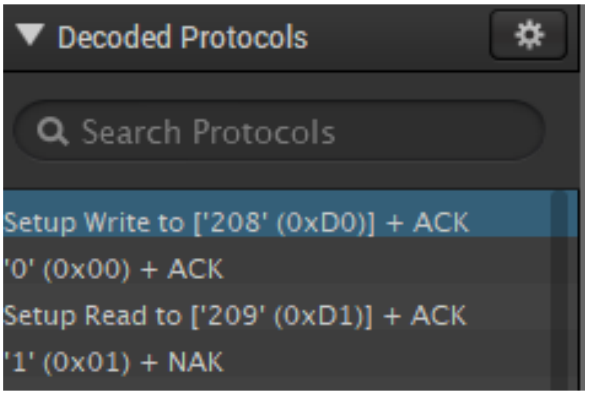
Implementoval: Alexander Valach Otestovala: Veronika Uhnáková

V nasledujúcej časti prikladáme záznamy z testovania systémových hodín.

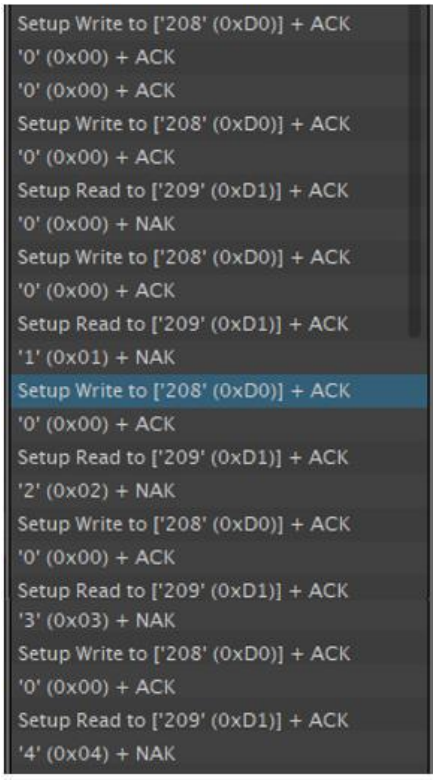


Obr. 13: Transakcia čítania 1. sekundy na logickej sonde (signál)

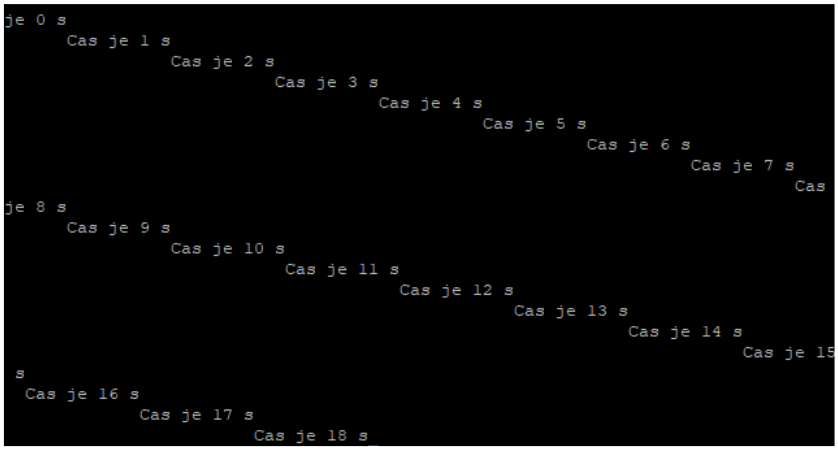
V nasledujúcej časti prikladáme záznamy z testovania systémových hodín. Obrázok 13 znázorňuje transakciu čítania prvej sekundy po spustení hodín na logickej sonde. Samotná transakcia je na Obrázku 14. Obrázok 15 znázorňuje opakovanú transakciu čítania jednej sekundy a Obrázok 16 vypísanie počtu sekúnd v programe Putty cez USART (9600 baudov). Výpis bol realizovaný každú sekundu.



Obr. 14: Transakcia čítania



Obr. 15: Opakovaná transakcia čítania sekúnd



Obr. 16: Zobrazenie počtu sekúnd cez USART

4.2.2.2. Test merania teploty

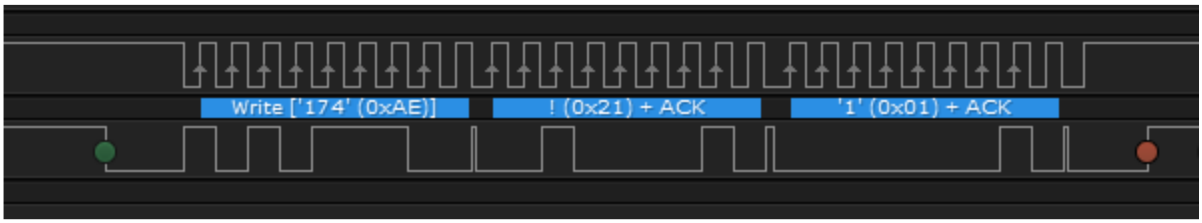
Implementovala: Veronika Uhnáková Otestoval: Alexander Valach

V nasledujúcej časti prikladáme záznamy z testov merania teploty. Obrázok 17 znázorňuje vykonanie jedného merania teploty, čomu predchádza spustenie merania zápisom hodnoty 1 do registra 21 (Obrázok 18). Obrázky 7 a 8 znázorňujú čítanie hodnôt z príslušných registrov 1F a 20, v ktorých sa nachádza hodnota nameranej teploty. Obrázok 19 znázorňuje

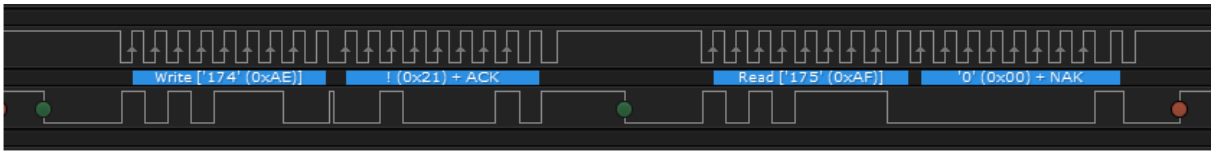
vypísanie nameranej hodnoty na USART cez program Putty (9600 baudov). Výpis bol realizovaný každú sekundu.

```
Setup Write to ['174' (0xAE)] + ACK
! (0x21) + ACK
'1' (0x01) + ACK
Setup Write to ['174' (0xAE)] + ACK
! (0x21) + ACK
Setup Read to ['175' (0xAF)] + ACK
'0' (0x00) + NAK
Setup Write to ['174' (0xAE)] + ACK
'31' (0x1F) + ACK
Setup Read to ['175' (0xAF)] + ACK
'28' (0x1C) + NAK
Setup Write to ['174' (0xAE)] + ACK
' ' (0x20) + ACK
Setup Read to ['175' (0xAF)] + ACK
'2' (0x02) + NAK
```

Obr. 17: Cyklus merania teploty

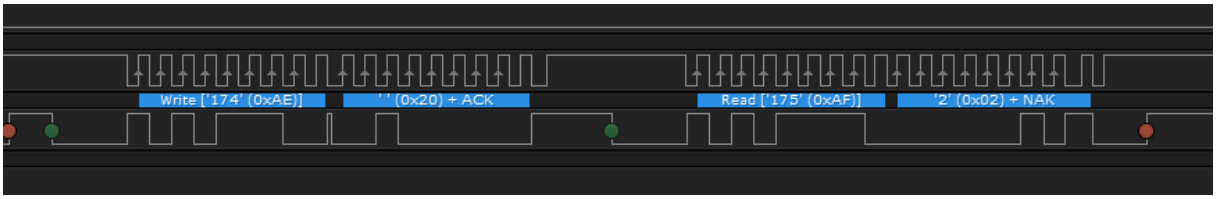


Obr. 18: Zápis hodnoty 1 do registra 21 - spustenie merania teploty

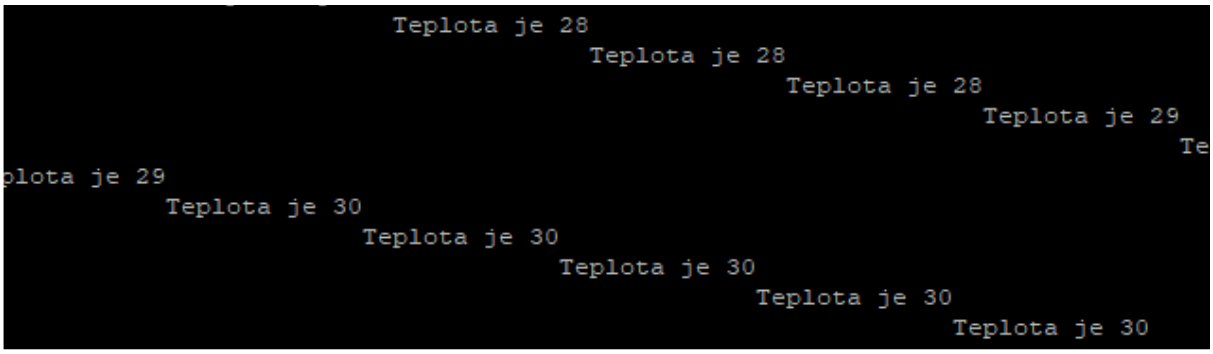


Obr. 19: Test, či už bolo meranie dokončené

Obr. 20: Čítanie nameranej hodnoty z registra 1F – Die Temp Integer



Obr. 21: Čítanie nameranej hodnoty - Die Temp Fraction



Obr. 22: Zobrazenie nameranej teploty v stupňoch Celzia cez USART

4.2.3. Bootloader

4.2.3.1. Pamäť programu

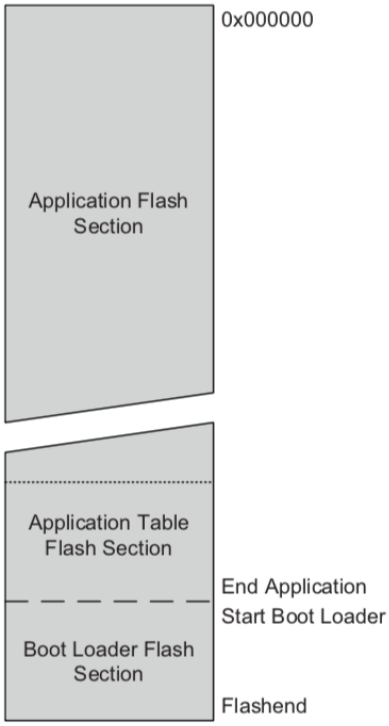
Flash pamäť je prístupná na zápis a čítanie pre externý programátor cez PDI alebo z aplikačného softvéru bežiacom v zariadení. Všetky inštrukcie AVR CPU majú šírku 16 alebo 32 bitov a každý blok vo flash pamäti je 16 bitový. Organizácia flash pamäte je rozdelená na dve sekcie, na aplikačnú sekciu a sekciu pre bootloader, ako možno vidno na (Obr. 23: Rozdelenie flash pamätí). Veľkostí sekcií sú fixné a záležia od zariadenia. Tieto dve sekcie majú separátne lock bity a môžu mať rôzne úrovne protekcie. Inštrukcia SPM (store program memory) používaná na zápis do flash pamäte z aplikačného softvéru, je vykonateľný iba z bootloader sekcie.

4.2.3.2. Aplikačná sekcia

Aplikačná sekcia je sekcia flash pamäte, ktorá je používaná pre vykonateľný aplikačný kód. Úroveň ochrany aplikačnej sekcie môže byť vybraná boot lock bitmi. V aplikačnej sekcii nemôže byť uložený bootloader, pretože inštrukcia SPM nemôže byť vykonaná z aplikačnej sekcie.

4.2.3.3. Bootloader sekcia

Kým aplikačná sekcia je použitá na uloženie aplikačného kódu, bootloader musí byť umiestnený v bootloader sekcii, pretože inštrukcia SPM môže spustiť programovanie, keď je vykonávaná z tejto sekcie. Inštrukcia SPM má prístup do celej flash pamäte vrátane bootloader sekcie. Úroveň ochrany bootloader sekcie môže byť nastavená bootloader lock bitmi. Ak sa táto sekcia nepoužíva pre bootloader, môže sa tu uložiť kód aplikácie.



Obr. 23: Rozdelenie flash pamätí

4.2.3.4. Návrh riešenia bootloadera

Keďže predpokladáme, že sa náš projekt bude časom zlepšovať, potrebujeme už pri písaní bootloadera myslieť na možnosť aktualizovania firmvéru na novú verziu. V pamäti dát sa bude vždy nachádzať posledná funkčná verzia firmvéru a nová verzia, ktorú bootloader musí nahrať do aplikačnej sekcie. Ak firmvér stiahne a overí novú verziu nastaví v EEPROM príznak pre bootloader, aby vedel že treba po resetovaní nahrať novú verziu do pamäte

4.3. Mobilná aplikácia

Mobilnú aplikáciu je dnes možné vytvoriť multiplatformovo alebo vyvíjať pre každú platformu samostatne. Dnes je veľmi populárny multiplatformový framework React-Native, ktorý umožňuje vytvárať aplikácie využitím JavaScriptu, ktorý je populárny medzi programátormi. Vďaka nemu je možné rýchlo a lacno vytvoriť aplikácie.

Pre mobilnú platformu iOS sa klasicky vytvárajú aplikácie pomocou programovacieho jazyka Swift.

Porovnanie React-Native s natívnymi aplikáciami.

Natívny vývoj:

- klady:
  - Široký prístup k API a knižniciam tretích strán
  - Možnosť implementovania natívnych UI elementov a zložitých animácií
  - Striktná typovosť
- zápory:
  - Vývoj aplikácie pre každú platformu
  - Zložitejšie a časovo náročnejšie vytvorenie aplikácie

React-native:

- klady:
  - Populárny multiplatformový framework
  - Hot reloading – rýchlejší debugovací proces
  - Ľahko dosiahnuteľný natívny výkon
  - Aktívne vyvíjaný a rastúci vďaka react-native komunite
  - Webový vývojári sa vedia rýchlo naučiť
- zápory:
  - Občas sa nájde chýbajúca knižnica tretích strán
  - Niektoré komponenty nie je možné vytvoriť v JavaScripte, potreba vývojára so skúsenosťami z natívneho vývoja
  - Zložené UI elementy a animácie zaostávajú za natívnou stranou

V projekte sme sa rozhodli ísť cestou natívneho vývoja z dôvodov držania kroku s mobilnými operačnými systémami a tiež tým, že React Native je abstrakciou a je zložitejšie identifikovať problém, ak narazíme na bug.

4.3.1. Návrh mobilnej aplikácie

Návrh pozostáva z 12-tich obrazoviek, ktoré predstavujú približný dizajn našich mobilných aplikácií. Vzhľad samotných obrazoviek sa môže ešte v priebehu času meniť.

Obr. 24 Návrh mobilných obrazoviek

Obrazovku číslo 1 tvorí prihlasovanie používateľa jeho e-mailom a používateľským heslom. Ak potenciálny používateľ ešte nemá vytvorené svoje konto, využije možnosť registrovať sa, ktorá ho presmeruje na obrazovku číslo 2.

Obrazovka číslo 2 je formulár s údajmi, ktoré je potrebné vyplniť pre úspešnú registráciu. Tri bodky znamenajú ešte ďalšie pole a to pole presne adresy – ulica, číslo domu, poštové číslo, mesto, krajina.

Obrazovka číslo 3 je hlavná obrazovka pre výber náramku Breyslet, ktorý chceme sledovať, resp. pozorovať namerané hodnoty nositeľa daného náramku. Taktiež je tu možnosť pridať nový náramok, ktorý chceme sledovať, ktorý nás následne naviguje na zadanie bezpečnostného PIN kódu, ktorý slúži na overenie, aby neoprávnená osoba nemohla sledovať cudzí náramok Breyslet.

Obrazovka číslo 4 zobrazuje posledné namerané hodnoty nositeľa daného náramku. Na tejto obrazovke bude nižšie tiež voľba pre zobrazenie poslednej zaznamenanej polohy.

Obrazovka číslo 5 obsahuje kontaktné a aj osobné informácie o nositeľovi náramku, vrátane krvnej skupiny.

Obrazovka číslo 6 obsahuje súbor možných nastavení, či už zmeny hesla, nastavenia jazyka aplikácie, zobrazení osobných informácií o používateľovi aplikácie a pre administrátorského používateľa tiež pridanie ďalších používateľov.

Obrazovka číslo 7 zobrazuje osobné informácie o používateľovi aplikácie.

Obrazovka číslo 8 slúži pre zmenu jazyka aplikácie.

Obrazovka číslo 9 umožňuje zmenu hesla používateľského účtu.

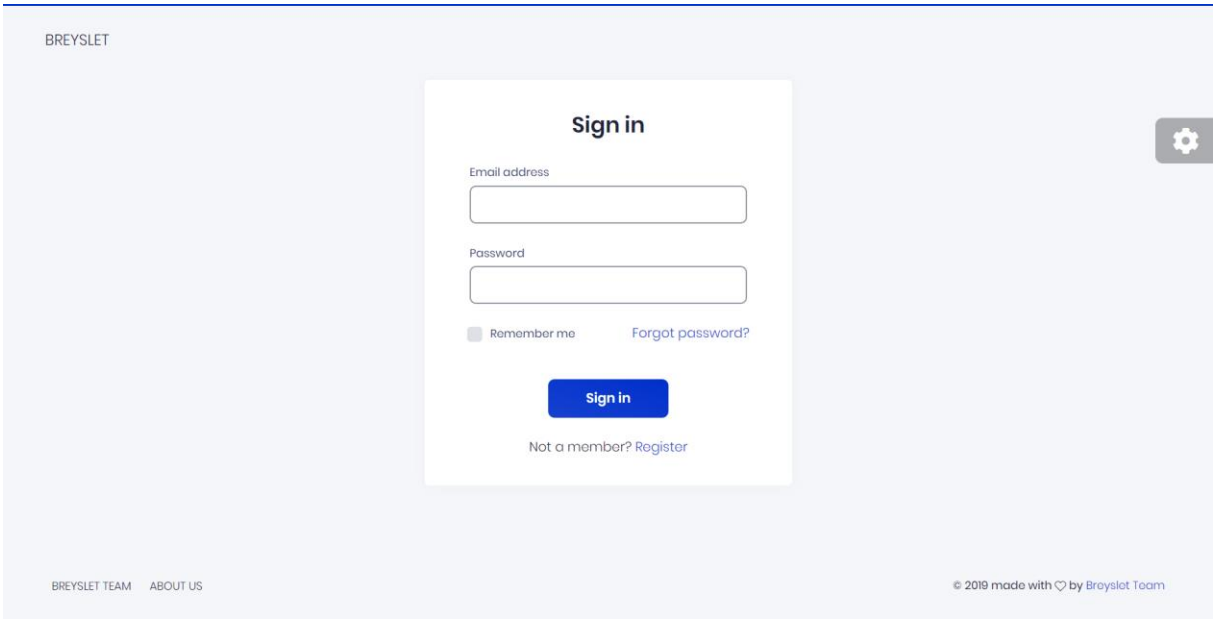
Obrazovka číslo 10 sa zobrazí po kliknutí na možnosť Events v dolnom navigačnom menu, kde sa zobrazia posledné notifikácie o stave nositeľa náramku spolu s nameranými hodnotami.

Obrazovka číslo 11 zobrazuje lekárske záznamy o nositeľovi náramku, názov diagnózy a krátky popis, ktorý je možné otvoriť na samostatnej obrazovke.

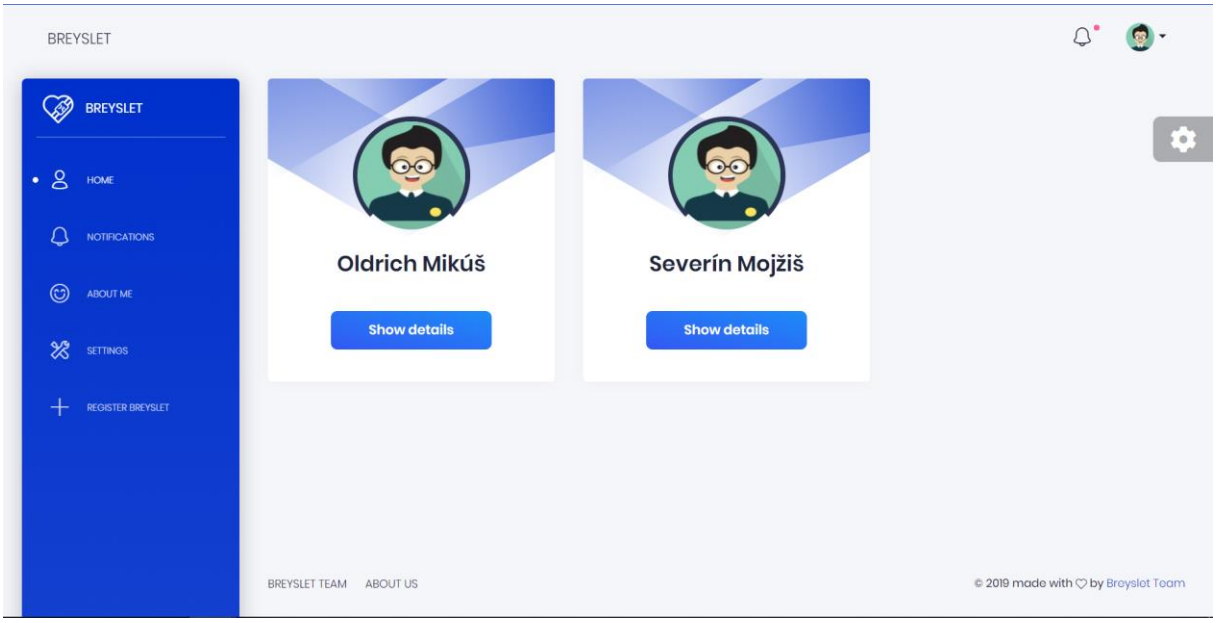
Obrazovka číslo 12 obsahuje neskrátený lekársky popis konkrétnej zvolenej diagnózy aj spolu s liečivami, ktoré užíva nositeľ náramku.

4.3.2. Webová aplikácia

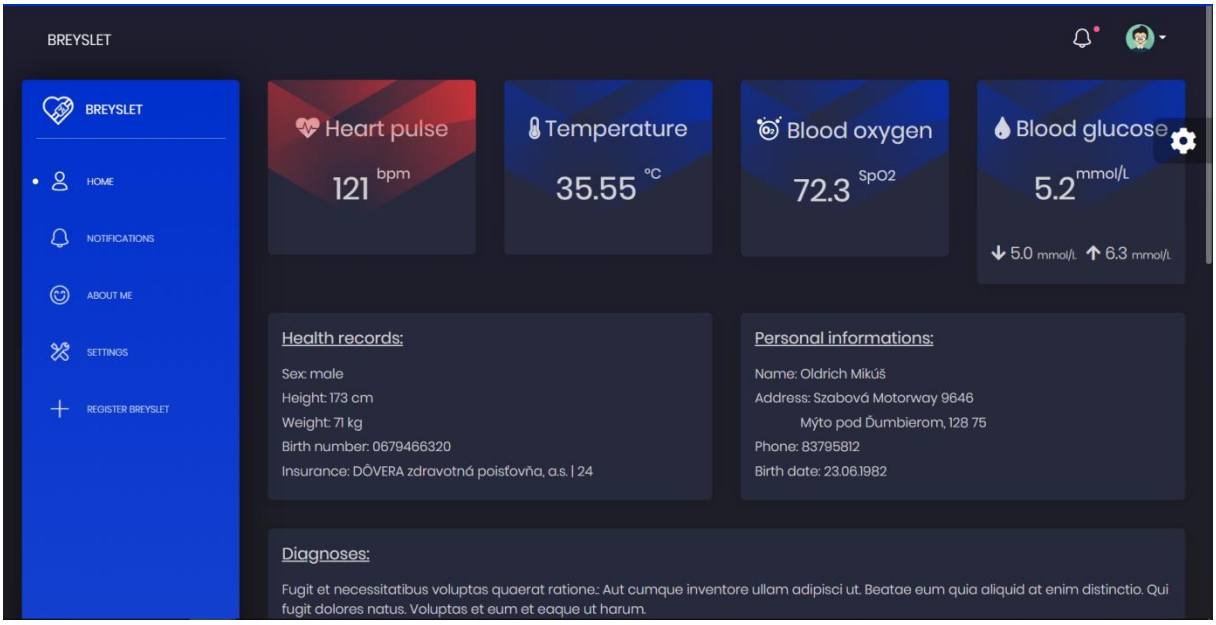
Webová aplikácia je postavená na PHP frameworku Laravel. Funkcionalita webovej aplikácie je ekvivalentná mobilným aplikáciám. Po prihlásení sa používateľovi zobrazí zoznam nositeľov náramku. Po kliknutí na používateľa sa zobrazí detail používateľa s aktuálne nameranými údajmi. Webová aplikácia podporuje možnosť zobraziť graf z naposledy nameraných hodnôt. Okrem toho sa zobrazí aj mapa s polohou nositeľa náramku. Webová aplikácia taktiež disponuje možnosťou registrovať si nový Breyslet a teda pridať si nového nositeľa náramku. V prípade, že už je nositeľ registrovaný, aplikácia umožňuje poslať pozvanie na zdieľanie informácií o nositeľovi s iným používateľom. Okrem toho aplikácia disponuje aj množstvom ďalších nastavení pre komfort používateľa, ako napríklad prepnutie svetlého a tmavého režimu. Nový používateľ sa vie veľmi rýchlo zaregistrovať.



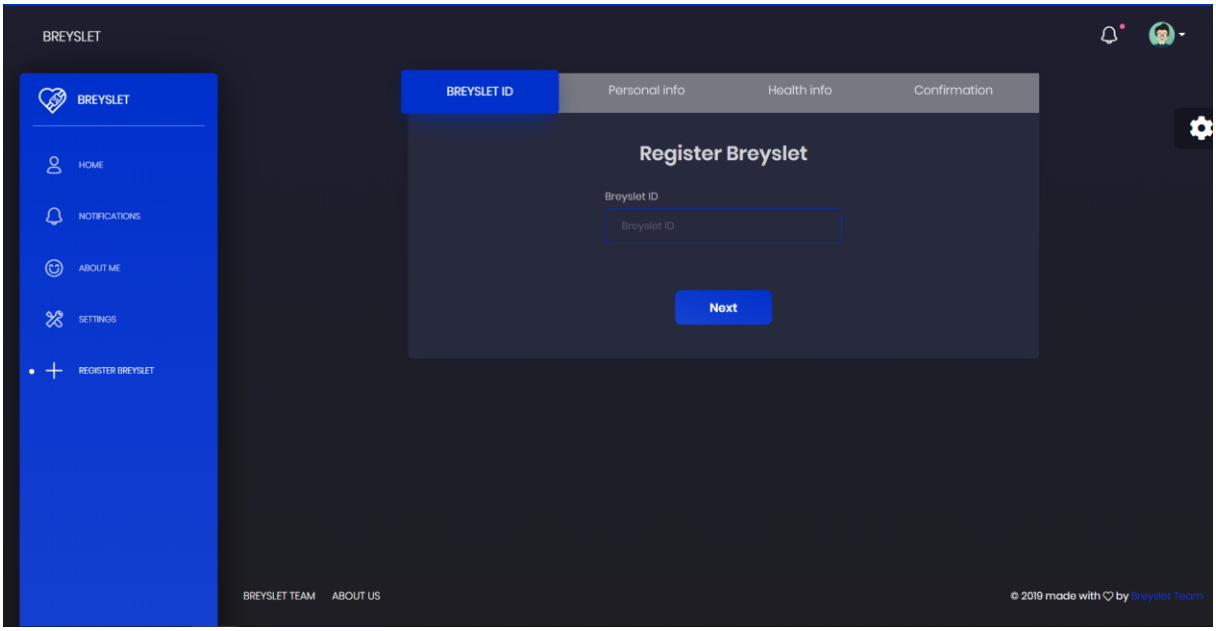
Obr. 25.: Webová aplikácia - obrazovka prihlásenia



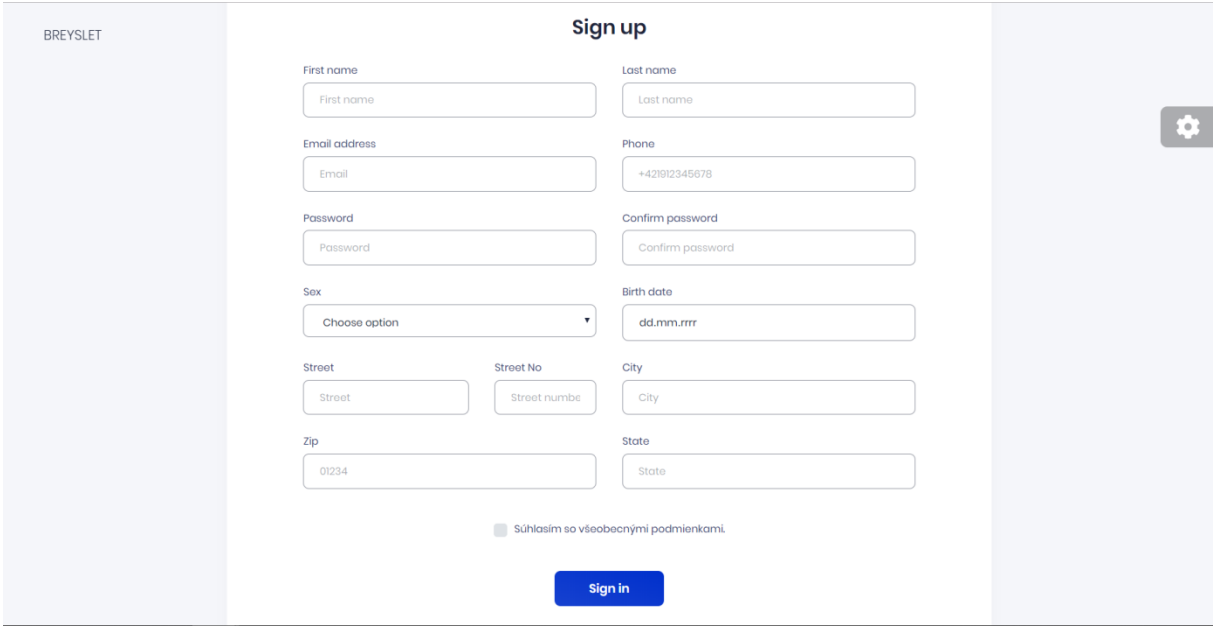
Obr. 26.: Webová aplikácia – obrazovka zoznam nositeľov Breysletov



Obr. 27.: Webová aplikácia – obrazovka detail nositeľa Breysletu (v tmavom režime)



Obr. 28.: Webová aplikácia – obrazovka registrácie nositeľa Breysletu (v tmavom režime)



Obr. 29.: Webová aplikácia – obrazovka registrácie používateľa

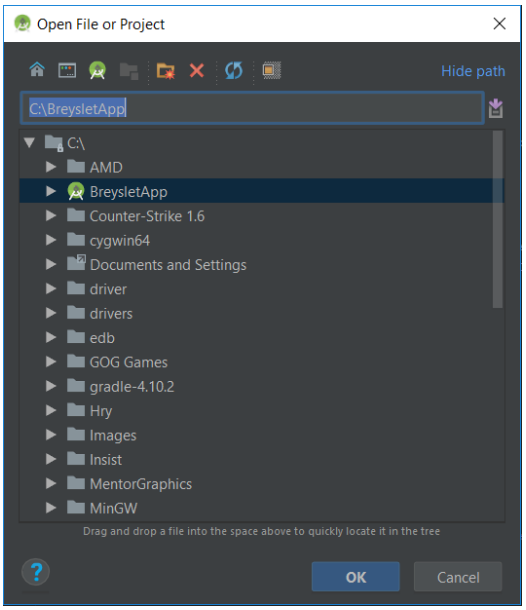
5. Príručky

5.1. Inštalačná príručka pre Android aplikáciu

Pre korektné spustenie vývojovej verzie mobilnej aplikácie pre systém android je potrebné stiahnuť si aktuálnu verziu zdrojového kódu z úložiska gitlab:

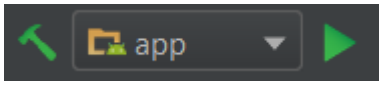
```
git clone git@gitlab.com:brave-foxes/breyslet-android-app.git
```

Použité a odporúčané vývojové rozhranie je Android Studio vo verzii 3.2 dostupné na webovej stránke https://developer.android.com/studio. Po nainštalovaní a spustení tohto rozhrania je možné otvoriť projekt kliknutím na položku menu v ľavom hornom rohu - File → Open a v dialógovom menu zvoliť priečinok, kde sa nachádza stiahnutý zdrojový kód.



Obr. 30.: Dialógové menu

Po otvorení projektu a inicializácií indexov a počiatočných nastavení je možné projekt skompilovať, prípadne spustiť (použitím vstavaného aplikačného emulátora alebo zapojením reálneho zariadenia s povolenou funkciou USB debugging), kliknutím na položku Make project/Run.



Obr. 31: Kompilácia/spustenie projektu

5.2. Inštalačná príručka pre IOS aplikáciu

V nasledujúcom návode je stručne zhrnuté ako spustiť, testovať a nainštalovať aplikáciu Breyslet.

5.2.1. Získanie projektu

Projekt sa nachádza v online gitlabe alebo na digitálnom nosiči.

```
$ git clone https://gitlab.com/brave-foxes/breyslet-ios-app.git
```

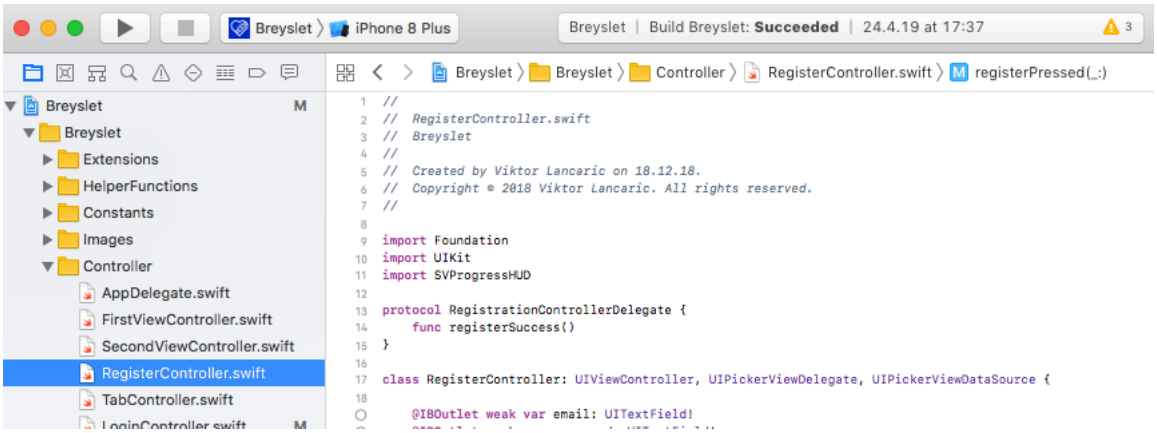
5.2.2. Spustenie a testovanie projektu

Projekt je potrebné otvoriť v Xcode štúdiu. Pred prvým otvorením je potrebné nainštalovať potrebné Cocoa Pods. V priečinku projektu spustíme príkaz:

```
$ pod install
```

A potom projekt otvoríme cez `Breyslet.xcworkspace`

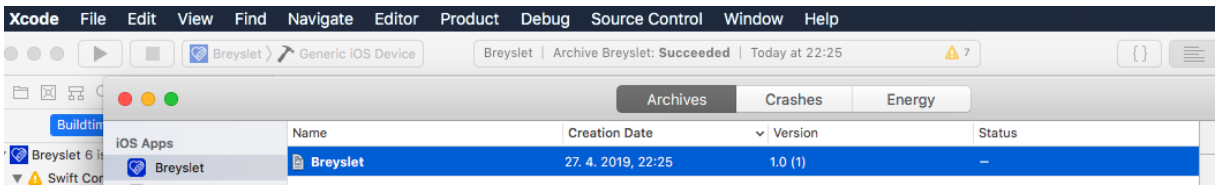
V Xcode si vyberieme aktívnu schému Breyslet a iOS simulátor alebo fyzické zariadenie a spustíme.



Obr. 32.: Spustenie projektu cez Xcode

5.2.3. Pushnutie aplikície do Testflightu/AppStoru

Najprv treba zmeniť target zariadenie na `Generic iOS device` . V hlavnom menu vyberieme Product -> Archive a počkáme kým sa vytvorí build aplikácie. Tento archív nahráme do App store connect a ďalej sprístupníme používateľom alebo testerom cez Testflight.

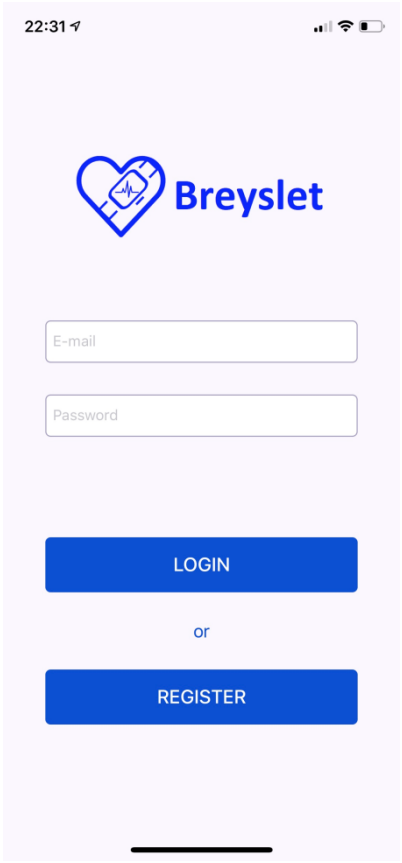


Obr. 33.: Vytvorenie archívu aplikácie

5.2.4. Spustenie aplikácie

Aplikáciu stiahneme z Testflight alebo z App store a spustíme.

Obr. 35.: Nainštalovaná aplikácia



Obr. 34.: Úvodná obrazovka s prihlásením

5.3. Inštalačná príručka pre server

Treba sa uistiť či váš server spĺňa nasledujúce požiadavky:

- PHP> = 7.1.3
- Rozšírenie PHP OpenSSL
- Rozšírenie PHP PDO
- Mbstring Rozšírenie PHP
- Rozšírenie PHP Tokenizer
- XML PHP rozšírenie
- Ctype PHP Extension
- Rozšírenie PHP JSON
- Rozšírenie PHP pre BCMath

1. Pred inštaláciou Laravel projektu nainštalujeme MySQL databázu vo verzii 5.7. https://dev.mysql.com/downloads/mysql/5.7.html
2. Laravel využíva Composer na správu svojich závislostí. Pred použitím Laravel sa uistite, že máte na počítači nainštalovaný Composer.
3. Projekt sa nachádza na gitlabe alebo na digitálnom nosiči. „git clone https://gitlab.com/brave-foxes/php-server.git“
4. Pred migráciou databázy je potrebné upraviť súbor .env, kde zadáme meno, heslo, adresu mysql servera.
5. Následne v priečinku s projektom spustíme príkazy:
   a) „composer up“
   b) „composer install“
   c) „php artisan migrate --seed“
   d) „php artisan passport:install“
   e) „php artisan key:generate“
6. Spustíme server príkazom „php artisan serve“

5.4. Inštalačná príručka webovej aplikácie

Treba sa uistiť či váš server spĺňa nasledujúce požiadavky:

- PHP> = 7.1.3
- Rozšírenie PHP OpenSSL
- Rozšírenie PHP PDO
- Mbstring Rozšírenie PHP
- Rozšírenie PHP Tokenizer
- XML PHP rozšírenie
- Ctype PHP Extension
- Rozšírenie PHP JSON
- Rozšírenie PHP pre BCMath

1. Laravel využíva Composer na správu svojich závislostí. Pred použitím Laravel sa uistite, že máte na počítači nainštalovaný Composer.
2. Pri webovom projekte sme použili aj node package manager. Preto je potrebné mať nainštalované https://nodejs.org/en/download/.
3. Projekt sa nachádza na gitlabe alebo na digitálnom nosiči. „git clone https://gitlab.com/brave-foxes/web-application-2.git“
4. Následne v priečinku s projektom spustíme príkazy:
   a) „composer up“
   b) „composer install“
   c) „npm install“
   d) „npm run dev“
   e) „php artisan key:generate“

Ak je potrebné zmeniť url adresu servera, tak to je možné vykonať v súbore app\Http\Controllers\ApiController.php prepísaním premennej base_uri v úvode súboru.

5. Spustíme aplikáciu príkazom „php artisan serve --port:8001“

Prílohy

A – Protokol o testovaní Tím 15

A- Protokol o testovaní Tím 15

Táto príloha obsahuje testovanie produktu Breyslet s prislúchajúcimi súčasťami tímom číslo 15. Testovanie prebehlo v priestoroch FIIT STU v učebni 5.44. Členom tímu 15 sme predviedli testovacie scenáre. Nižšie je priložená správa o testovaní od tímu 15.

Priebeh testovania

Tím Breyslet nám previedol výsledky svojej práce na tímovom projekte. Vytvorili systém na monitorovanie osôb. Systém pozostáva z náramku, Breyslet, mobilných aplikácií pre operačné systémy, Android a iOS, webovej aplikácie a backendu. Tím vytvoril vlastný hardvér a vlastný firmvér.

Testovací scenár pre hardvér:

1. Priloženie ukazováka k senzorom
2. Stlačenie SOS tlačidla
3. Kontrola zobrazenia notifikácií v aplikáciách

Testovací scenár pre aplikácie a webovú stránku:

1. Zaregistrovanie používateľa
2. Zobrazenie sumárnych údajov o nositeľovi Breysletu
3. Zobrazenie detailu ľubovoľného parametra o nositeľovi

Zhodnotenie testovania

Testovanie scenára pre hardvér prebehlo bez problémov. Tím Breyslet nám pripravil prototyp zariadenia. Po stlačení tlačidla zariadenie viditeľne meralo parametre z prsta. Chvíľu nato sa v aplikácii a webovej stránke zobrazila notifikácia o stlačení tlačidla. Notifikácia na iOS zariadení sa nezobrazila, pretože tím Breyslet nevlastnil platený vývojársky účet Apple. Pri testovaní sme museli byť veľmi opatrní pri práci s prototypom, bol veľmi krehký, pospájaný káblikmi.

Aplikácie sme si nemohli nainštalovať na vlastné zariadenia, keďže neboli dostupné na obchode Google play alebo AppStore. Aplikácie pôsobili príjemným dojmom. Dizajn webovej stránky a aplikácií sa podobal. Obe aplikácie boli intuitívne. Jednoducho sme si zobrazili prehľad a detailné informácie o nositeľovi. Na iOS aplikácii ešte nebolo implementované zobrazenie grafu. Okrem toho sme nenašli žiadny problém čo sa týka dizajnu alebo funkčnosti aplikácií.

Webová stránka bola tiež v poriadku. Veľmi sa nám páčila možnosť zmeny farebného dizajnu stránky na tmavý. Registrácia bola zdĺhavá, bolo potrebné vyplniť veľa údajov. Po registrácii prišiel mail a konto bolo aktivované. Po stlačení tlačidla sa na webe tiež zobrazila varovná hláška.

Zhodnotenie

Náramok ešte nie je pripravený na nasadenie do praxe. Myslíme si, že tím Breyslet odviedol kus roboty. Celkový dojem z riešenia hodnotíme ako pozitívny. Navyše, žiaden člen tímu Breyslet nemal predošlé skúsenosti s programovaním firmvéru alebo návrhom hardvéru.