ETDK-dolgozat

Both Bence

Lukács István

XXVI. reál- és humántudományi Erdélyi Tudományos Diákköri Konferencia (ETDK)

Informatika II.: innovatív számítástechnikai termékek, alkalmazások szekció

Kolozsvár, 2023. május 18–21.

Sensor Throne

Ülési testtartás javítása és megfigyelése telefonos alkalmazással

Szerzők:

Both Bence

Babeş–Bolyai Tudományegyetem, Matematika és Informatika Kar, Informatika szak, III. év

Lukács István

Babeş–Bolyai Tudományegyetem, Matematika és Informatika Kar, Informatika szak, III. év

Témavezetők:

dr. Sulyok Csaba, egyetemi adjunktus

Babeş–Bolyai Tudományegyetem, Matematika és Informatika Kar

Brassai Beáta, szoftverfejlesztő,

Codespring

Szabó Zsolt, szoftverfejlesztő,

Codespring

CODESPRING

OUR CODE YOUR SUCCESS

Kivonat

A Sensor Throne projekt keretén belül egy okos szék került megvalósításra, amelynek célja a helyes ülési testtartás monitorizálása, illetve javítása. A rendszer két részt foglal magába, egy hardware és egy software komponenst. A hardware komponenst egy irodai székre szerelt mikrokontroller és a hozzá tartozó perifériák (akkumulátor, szenzorok, Bluetooth modul) alkotják, a software komponens pedig egy mobil alkalmazásból áll.

A mikrokontrollerrel az applikáció Bluetooth kapcsolaton keresztül, egy saját szöveges protokoll segítségével kommunikál. Az alkalmazás interaktív felületet nyújt a felhasználó számára, ahol láthatja a pillanatnyi testtartását, az adott széken való üléssel kapcsolatos statisztikáit, valamint személyre szabhatja az alkalmazás beállításait.

Az alkalmazás jelenleg három nyelven érhető el, amely a telefonon beállított nyelv alapján automatikusan kiválasztásra kerül. A felhasználó értesítést kap a helytelen vagy túl sokáig tartó ülés esetén, illetve ha a mikrokontrollert vezérlő akkumulátor feszültsége alacsony.

A dolgozat további része a követelményekről, a funkcionalitásokról és az architektúráról ad leírást, illetve ábrák segítségével demonstrálja az egység helyes működést.

XXVI. ERDÉLYI TUDOMÁNYOS DIÁKKÖRI KONFERENCIA
CONFERINŢA ŞTIINŢIFICĂ STUDENŢEASCĂ DIN TRANSILVANIA (CŞST), EDIŢIA A XXVI-A
26TH TRANSYLVANIAN STUDENTS' SCIENTIFIC CONFERENCE (TSSC)
reál- és humántudományok • ştiinţe reale şi umane • real and human sciences

KOLOZSVÁR, 2023. MÁJUS 18–21.

Kolozsvári Magyar Diákszövetség • Uniunea Studenţească Maghiară din Cluj
400083 Kolozsvár, Petőfi Sándor/Avram Iancu utca 21 • tel./fax: 0755116251
e-mail: *etdk@kmdsz.ro, etdk.kolozsvar@gmail.com* • honlap/web: *https://etdk.kmdsz.ro*

Témavezetői igazolás

Alulírott *dr. Sulyok Csaba, Brassai Beáta és Szabó Zsolt* igazoljuk, hogy *Both Bence és Lukács István* hallgatók a(z) *Sensor Throne: Ülési testtartás javítása és megfigyelése telefonos alkalmazással* című dolgozatot az alulírottak szakmai irányításával készítették el, és javasoljuk az említett tudományos munka bemutatását a XXVI. Reál- és Humántudományi Erdélyi Tudományos Diákköri Konferencia (ETDK) *Informatika II.: innovatív számítástechnikai termékek, alkalmazások* szekciójában.

dr. Sulyok Csaba, egyetemi adjunktus
Babeş–Bolyai Tudományegyetem, Matematika és Informatika Kar

Brassai Beáta, szoftverfejlesztő,
Codespring

Szabó Zsolt, szoftverfejlesztő,
Codespring

Kolozsvár, 2024. január 4.

Tartalomjegyzék

1. Bevezető

Az elmúlt időszakban számos felmérés kimutatta, hogy jelentősen megnövekedett az ülőmunkák száma. Európa lakosságának már 39%-a végez ülőmunkát [19], míg az amerikai kontinensen 1950-hez képest 83%-ot növekedett [18] azon munkák száma, amelyeket ülő pozicióban végeznek. Így az amerikaiak jelenleg többet ülnek életük során, mint bármikor a történelemben. Az említett életmód miatt az emberek hajlamosabbak egészségtelenebbül ülni, ezért egyre több betegség éri az irodai munkásokat: lassul az anyagcsere, gerinc problémák, mozgásszervi panaszok, emésztőrendszeri panaszok jelentkezhetnek [6].

A Sensor Throne nevű projekt célja egy olyan szék készítése, amely segít a felhasználójának ülési szokásait monitorizálni, illetve ülési testtartását javítani. Mindezt egy telefonos alkalmazás, egy székre szerelt mikrokontroller és az ehhez tartozó szenzorok segítségével valósítja meg. A mikrokontroller illetve az alkalmazás közötti kommunikáció Bluetooth segítségével megy végbe. Az alkalmazás látványos grafikonok segítségével ad betekintést felhasználója ülési szokásaiba.

A jelenlegi projekt előtt, már volt egy probálkozás egy csapat részéről, bár nem bizonyult túl eredményesnek. Nekik sikerült megvalósítani egy hardware komponens összerakását, ahol egy Particle Photon mikrokontrollert használtak, a jelenlegi projekthez hasonló módon szenzorokat építettek be a székbe. Továbbá ehhez párosítottak egy web alkalmazást, ahol nyomom lehet követni az ülési testtartást.

Az próbálkozás esetén az okos szék Wi-Fi segítségével kommunikált egy web szerverrel. A hardware komponens felelős volt a szenzor adatok méréséért és rögzitéséért és ezek küldéséért a web alkalmazás fele. Mint utólag kiderült, a szenzorok nem mértek pontos értékeket és instabilan működtek, ezért a SensorThrone projekt során más szenzorokat kellett használni. Mivel az adatküldés Wi-Fi kapcsolaton keresztül történt, a kommunikáció nagyon költséges volt energiaellátás szempontjából, így a mikrokontrollerhez kapcsolt akkumulátor nagyon hamar lemerült. A web alkalmazás sem bizonyult a legmegfelelőbb választásnak, mivel a mai világban a telefonos alkalmazással egyszerűbben és gyorsabban el lehet érni az adatokat. Kimutatások támasztják alá, hogy a felhasználók telefonuk használatának 90%-át alkalmazások kezelésével töltik a böngészőben való keresgélés helyett [4].

Már megjelentek más okos székek is a világpiacon, ezek közül a legtöbb prototípus verzió. Akad már termékként szereplő okos szék is, például a Gregory Smart Chair, amihez elérhető alkalmazás úgy android, mint iOS eszközökön [13].

A dolgozat négy fő részből áll. Az első részben a projekt összetételéről van szó, a második részben a felhasznált technológiák vannak megemlítve, a harmadik részben a fejlesztői eszközök és a felhasznált módszerek vannak tárgyalva, illetve a negyedik részben a továbbfejlesztési lehetőségek vannak részletezve.

A projekt a 2022-es Codespring által szervezett szakmai gyakorlat alatt indult [8]. A nyári gyakorlat alatt a szenzorok behelyezésén illetve az alkalmazás megvalósításán volt a hangsúly. Sikerült megvalósítani a kommunikációt a telefon illetve a szék között, az alkalmazás oldalon létrehozni a fő nézeteket, mint például egy üléssel kapcsolatos kördiagrammot vagy a valós idejű monitorizálást.

A csoportos projekt tantárgy keretén belül folytatódott a projekt új csapattársakkal kiegészülve, név szerint: Márton Krisztián, Finta Zoltán, Dósa Csaba és Fazakas Hunor. A csapatmunka egy demo mód bevezetésével kezdődött, ami a hardware komponensnek egy előre meghatározott viselkedését szimulálja. Az ő segítségükkel készült el az alkalmazás többnyelvűsítése, a saját kommunikációs protokoll, illetve a beállítások nézet. Ezen kívül új diagrammok is készültek, melyek közül az egyik a fentebb említett üléssel kapcsolatos statisztikákat mutatja napi, heti illetve havi felbontásban, a másik pedig az akkumulátor használattal kapcsolatos statisztikákat jeleníti meg.

Köszönet illeti az Antares [2] székgyártó céget, amely biztosította az ötletet, a projekt fejlesztése során felhasznált széket illetve a pontos specifikációt. Továbbá köszönet a Codespring szoftverfejlesztői cég csapatának a projekt mentorálásáért, a menotoroknak: Brassai Beátának és Szabó Zsoltnak és a témavezető tanárnak, Sulyok Csabának.

2. A projekt komponensei

Ebben a fejezetben a projektet alkotó két nagy részről lesz szó. Kifejtésre kerül a projekt hardware része, mely a székből illetve az arra szerelt komponensekből áll illetve a szoftver rész is, melyet egy telefonos alkalmazás alkot.

2.1. Hardware komponensek

2.1.1. Miért erre a mikrokontrollerre esett a választás?

Annak érdekében, hogy a megfelelő mikrovezérlővel legyen ellátva a szék, ennek kiválasztásakor több szempontot is figyelembe kell venni. Az egyik szempont az volt, hogy a szék hordozható kell legyen, ami azt jelenti, hogy szükség volt egy akkumulátor beszerelésére is. Azért, hogy az akkumulátort minél kevesebbszer kelljen feltölteni, egy kis energiafogyasztású mikrovezérlőt kellett választani. A kisebb energiafogyasztás eléréséhez az applikáció és az elektronikai komponensek között zajló kommunikációt is optimalizálni kellett. A Bluetooth és a Wi-Fi eszközök közül a Bluetoothra esett a választás, mivel ennek az energiaigénye jóval alulmarad a Wi-Fi energiaigényével szemben [24]. Egy másik szempont volt a mikrovezérlő ára. Ez azért volt fontos, mert a szék önmagában is költséget jelent. Nem lett volna gazdaságos egy drága mikrovezérlő beszerelése, hogyha egy olcsó is megfelel. Ez a tömeggyártásra nézve nagyon fontos.

A projekt kezdetén több mikrovezérlő is összehasonlításra került az előbb említett szempontok szerint. Ezek név szerint a következők: Arduino UNO, Particle Photon, ESP 32, Raspberry Pi Pico. Végül a Raspberry Pi Picora esett a választás az 1. összehasonlító táblázat alapján.

Amint az a 1. táblázatban is látható, az Adruino Uno gyártása már befejeződött, nincs beépített Wi-fi és Bluetooth modulja illetve teljesítménye látványosan elmarad a többi mikrovezérlő teljesítményétől. A Particle Photon esetében is hiányzik a beépített Bluetooth modul és csak egy webes felületen keresztül lehet programozni, amely korlátozza a fejlesztők munkáját és ez elég nagy hátrány. Az ESP 32 és a Raspberry Pi Pico összehasonlítása után a Picora esett a választás. Bár a Raspberry Pi Pico teljesítménye kicsit elmarad az ESP 32 teljesítményétől, a választás mégis erre a mikrovezérlőre esett, mivel jóval jutányosabb áron lehet beszerezni. A Raspberry Pico teljesítménye elégséges a székre szerelt szenzorok működtetéséhez és a kommunikáció lebonyolításához. Egy másik előnye, hogy könnyen lehet

	Arduino UNO	Particle Photon	ESP 32	Raspberry Pi Pico
CPU Frekvencia	16 Mhz	120 Mhz	240 Mhz	133 Mhz
Flash memória	32 kb	1 MB	4 MB	2MB
Bluetooth	nincs	nincs	van	van
Wi-fi	nincs	van	van	van
Megjegyzések		webes felület	egy magos	kibővíthető modulokkal
Gyártási élettartam	Nem gyártják	A gyártását leállítják	Esp32 c-series 2033 januárjáig	Legalább 2028 januárjáig
Ár	12$	20$	10$	6$

1. ábra. Mikrokontrollerek összehasonlítása

hozzá kiegészítőket kapni mint például az UPS modul, mely árammal látja el a mikrovezérlőt.

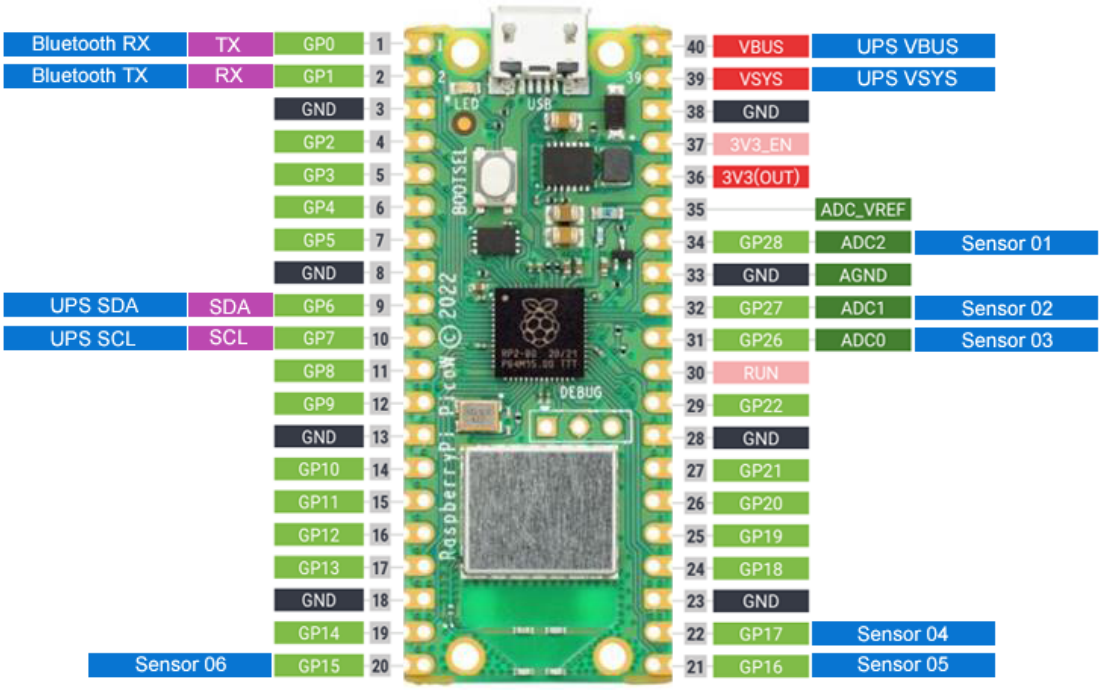
2.1.2. Összeszerelési folyamat

A székre szerelt egység több hardver modulból áll. Mivel a Raspberry Pi Piconak [16] (későbbiekben csak Pico) a beépített Bluetooth modulja még nem volt támogatott a projekt összeszerelési ideje alatt ezért szükség volt egy külső Bluetooth modul beszerzésére, amit a Picohoz lehetett csatlakoztatni.

A kiválasztott Bluetooth modul a HC-05 lett. A modul és a mikrovezérlő közötti kommunikáció UART protokoll [10] alapján működik. Ez az egyik legtöbb helyen használt kommunikációs protokoll. Az adatok továbbítása bitenként történik egy kábelen keresztül. Mivel a mikrovezérlő és a Bluetooth modul közötti kommunikáció kétirányú, ezért mindkét eszköz küldő illetve fogadóként is funkcionál. Ez azt jelenti, hogy a küldő (Transmitter-TX) jelet össze kell kötni a fogadó (Receiver-RX) jellel. Ez az összekötés jól látszik az 2. ábrán.

Miután a kommunikációhoz szükséges elemek a helyükre kerültek az áramellátáson volt a sor. A Picohoz egy Pico UPS A [23] nevű UPS modul lett hozzákötve, ami egy akkumulátor segítségével nyújtja a szükséges áramellátást. A kommunikáció az I2C [5] protokoll segítségével megy végbe a modul és a mikrovezérlő között. A kommunikációban részt vevő eszközök master és slave funkciókat kapnak. A kommunikáció két szál segítségével történik. Az egyik szál az SDA (Serial Data) melyen az üzenetek küldése és fogadása történik. A másik szál az SCL (Serial Clock) amely az adatcsere szinkronizálásáért felelős. Ennek a protokollnak a segítségével történik az akkumulátor töltöttségi szintjének a monitorizálása.

A Bluetooth modulon és az UPS modulon kívül hat darab szenzor is hozzá van kötve a



2. ábra. Raspberry Pi Pico bekötési diagram [9]

Picohoz, amelyek a felhasználó ülési pozíciójának a megfigyeléséért felelősek. A hat szenzor a Pico hat különböző pinjére lett kötve.

2.1.3. Az akkumulátor kapacitás meghatározása és tesztelése

A mikrovezérlőt egy UPS modul és egy hozzá kötött 3.7V elektormotoros feszültséggel és 800 mAh teljesítménnyel rendelkező akkumulátor működteti. Ez az akkumulátor csak teszt környezetben volt használva, de a későbbiekben tervbe van egy nagyobb kapacitású akkumulátor bevezetése is. Erre azért van szükség, hogy a szék ne egyfolytában egy áramforrás közelében kelljen legyen, hanem szabadon lehessen használni bárhol hosszabb ideig.

Annak érdekében, hogy meg lehessen becsülni az akkumulátor működési idejét több teszt is végre volt hajtva. Az akkumulátor tesztelve volt úgy, hogy, a mikrokontroller csak az akkumulátor töltöttségi szintjét figyelte. Így a becsült működési idő 36,8 óra volt. Második teszteset alatt a mikrovezérlőhöz hozzá volt kapcsolva a Bluetooth modul is. Így az akkumulátor 29 órán át szolgáltatott energiaellátást. Az utolsó teszteset mérései közben a teljes felszerelés üzemeltetve volt. Ekkor a hardware komponens intenzíven kommunikált a hozzá csatlakoztatott mobil alkalmazással. Az így mért maximális üzemeltetési időtartam 11 óra volt.

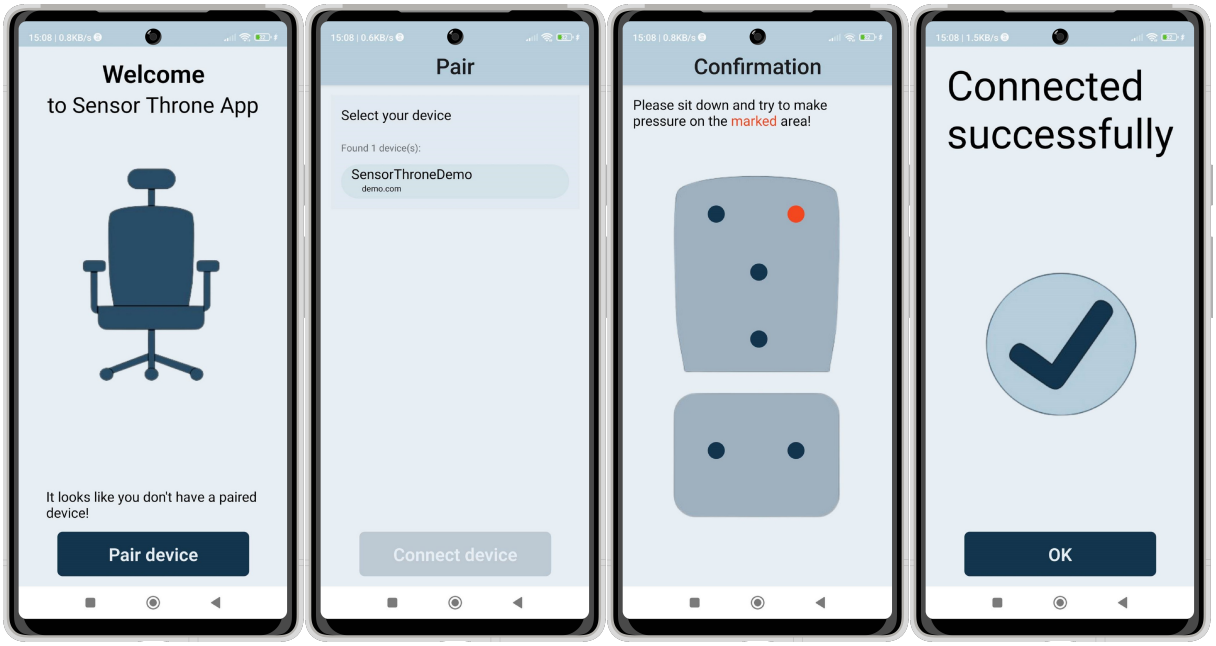
A mérések utáni következtetés a következő: a 11 órás akkumulátor üzemidő nagyon rövid ezért szükség van optimalizációra. Ezt egy úgynevezett „deep sleep" funkcionalitással lehetne elérni, mely nagyban megnövelné az üzemidőt. Erről a funkcionalitásról bővebben a 5 fejezet alatt lehet olvasni. Ezenkívül egy nagyobb kapacitású akkumulátor beszerzése is segítené az üzemidő kitolását.

2.1.4. Felmerülő problémák

Számos probléma merült fel a hardware komponens összeállítása és müködtetése közben. Az első probléma a szenzorok terén merült fel. Amint az bevezetőben is említve volt, már volt egy próbálkozás egy ilyen szék elkészítésére, viszont az akkor használt szenzorok instabilan működtek. Alaphelyzetben a szenzorok jól működtek, helyes értéket adtak vissza, viszont amikor megérintette valaki őket, akkor pontatlan értékeket térítettek vissza. Ezért szükséges volt ezeknek a szenzoroknak a kicserélése, így TTP223 [21] touch szenzorok kerültek a helyükre. Ezek beépítése után, tesztelés közben derült ki, hogy a kiválasztott szenzor típus 6-20 másodpercenként újrakalibrálja magát. Ez azt jelenti, hogy kalibráláskor a szenzor újra beállítja az offset értékét. Tehát a kalibrálás után a szenzort újra meg kell érinteni ahoz, hogy megfelelő jelet küldjön. Ezért ezek az érzékelők sem bizonyultak megfelelőnek. Végül mechanikus microswitchek kerültek a székbe. Az újonnan kiválasztott szenzorok nagyobb helyet foglalnak, mint egy előzőleg használt kapacitív szenzor, emiatt a székbe való beszerelésük problémásabb volt.

Egy másik nehézséget a mikrokontroller Bluetooth modul szoftverének támogatottsága okozott. A fejlesztés kezdetekor a Raspberry Pi Pico W mikrokontrollerhez nem publikáltak olyan firmware-t, ami segítségével használni lehetett volna a beépített Bluetooth modult. Ezért egy külső Bluetooth egységet kellett hozzákötni a hardware komponenshez, és ennek működését megoldani szoftver részen. Az előbb említett belső modul támogatása 2023 februárjában jelent meg, a hozzá tartozó példaprogramokkal [26]. Hosszútávú tervek közé tartozik a beépített modul használatba helyezése.

A mikrokontroller két maggal rendelkezik, így két szálon futó programot lehetséges elkészíteni. Ahhoz, hogy minden funkcionalitás időben végrehajtódjon, a mikrokontroller szoftverének struktúrája és működése megfelelő tervezést igényelt.



(a) (b) (c) (d)

3. ábra. Onboarding nézetek

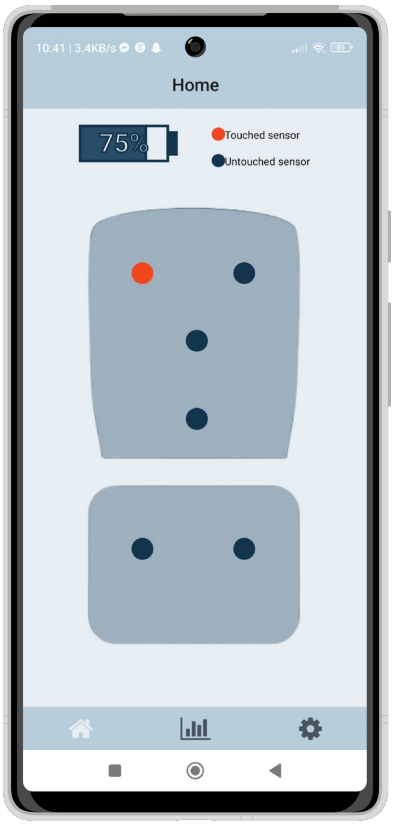
2.2. Mobil alkalmazás

2.2.1. Főbb funckionalitások

Onboarding

Az alkalmazás első használatának a feltétele a telefon és a szék közötti kommunikáció felépítése. Ez a kapcsolat Bluetoothon keresztül valósul meg, ezért a felhasználónak párosítania kell telefonját a székhez. Annak érdekében, hogy az alkalmazás biztosan a kívánt székkel kapcsolódjon össze illetve a felhasználó dolgának megkönnyítésére, a fejlesztők létrehoztak egy onboarding folyamatot.

Az onboarding folyamat 4 lépésből áll és automatikusan jelenik meg a felhasználónak, amikor első alkalommal akar a telefonjával okos székhez kapcsolódni. Az első nézet, ahogy az a 3a. ábrán is látható egy üdvözlő üzenetet tartalmaz, amelyről egy gomb megnyomásával léphet a felhasználó tovább. A következő nézeten a közelben levő, elérhető okos székek listája jelenik meg. Ezek közül kiválasztható a használni kívánt szék, ez látható a 3b. ábrán. Minden széken el lesz helyezve egy cimke vagy egy QR kód, amely alapján a felhasználó a szék adataihoz hozzáférhet. Innen kaphatja meg a szék nevét, amit aztán a nézeten megjelenő listából kiválaszthat. A kiválasztást egy megerősítési folyamat követi, mely a felhasználót biztosítja arról, hogy a megfelelő székhez csatlakozott, illetve hogy a székbe beépített szenzorok



4. ábra. Valós idejű megfigyelés

megfelelően működnek. Ez a folyamat abból áll, hogy a széken elhelyezett szenzorokat a telefon képernyőjén megjelenő elhelyezkedési sorrendben kell megérinteni (3c. ábra). Miután ez megtörtént az 3d. ábrán megjelenő nézet biztosít arról, hogy sikeresen kapcsolódott a két eszköz és az OK gomb megnyomása után az alkalmazás készen áll a használatra.

Valós idejű megfigyelés

Az alkalmazás főnézete egy széket ábrázol, ahol megjelennek a székbe szerelt szenzorok fizikai pozíciói. Ez a nézet felelős a valós idejű megfigyelésért. A székre való leülés esetén a telefon kijelzőjén látható, hogy éppen melyik szenzor aktív. Azok a szenzorok, amelyek éppen meg vannak érintve, narancssárga kitöltést kapnak a kijelzőn, amint azt a 4. ábra is mutatja. Ezáltal a felhasználó hamarabb észreveszi, ha ülési pozíciója rossz irányba változik mivel a narancssárga kitöltés eltűnik. Ennek a funkcionalitásnak köszönhetően a felhasználó láthatja, hogy éppen mennyire ül helyesen a széken, illetve mi az amin változtatnia kellene.

(a) Napi statisztika (b) Heti statisztika (c) Éves statisztika

5. ábra. Statisztika nézetek

Statisztikák

A statisztika funkcionalitás az alkalmazás egyik legfontosabb része. Egyik előnye, hogy a felhasználó nem kell folyamatosan a valós idejű monitorizálást nézze a telefonján annak érdekében, hogy megfigyelje az ülési szokásait. Nyugodtan bezárhatja az alkalmazást és később visszatérve megtekintheti, hogy milyen mértékben ült helyesen. Másik előnye, hogy a felhasználó ezáltal figyelemmel követheti saját ülési szokásainak a változását is az idő múlásával.

Az alkalmazásban lehetőség van többféle statisztika megtekintésére, kördiagram illetve oszlopdiagram formájában is. Ezek a nézetek a képernyő alján levő grafikon ikon megnyomásával érhetőek el. Itt a felhasználó több különböző idő felbontásban is megtekintheti az ülési szokásait beleértve a történelmi adatokat is.

A fejlesztés során szükség volt a helyes ülés meghatározására a szenzor adatok függvényében: az alany akkor ül helyesen a széken, ha minden szenzor érintve van. Ha ezek közül csak néhány aktív, akkor helytelen testtartásról beszélhetünk. Ha egy szenzor sem aktív, akkor a szék nincs használatban.

A statisztika nézetre való navigáláskor a 5a. ábrán látható képernyőkép jelenik meg, ahol egy kördiagram mutatja a napi ülési statisztikát. Három szín válik láthatóvá: a narancssárga a helytelen, míg a sötétkék a helyes ülési testtartás alatt eltelt időt jeleníti meg. A világoskék szín a használaton kívűli időt ábrázolja. Ezek magyarázata is megtalálható a diagram alatt. Ugyancsak a diagram alatt folyamatosan változó érdekességek jelennek meg az üléssel kapcsolatban.

A felső navigáció segítségével kiválasztható egy korábbi dátum, ami által visszamenőleg is megtekinthetők a testtartással kapcsolatos diagramok. A dátum kiválasztása mellett, lehetőség van átnavigálni egy másik nézetre, ahol oszlopdiagramok segítségével látható heti, hónapos, illetve éves leosztásban az ülés nyilvántartása.

Amint a 5b. és a 5c. ábra is mutatja, az adatok oszlopdiagram formájában jelennek meg. A diagramok azt mutatják, hogy hány százalékban volt a testtartás helyes, illetve helytelen az elmúlt időszakban.

Az említett diagramok frissítése „pull to refresh" módszerrel történik, amiről az 2.2.1 fejezetben van szó bővebben.

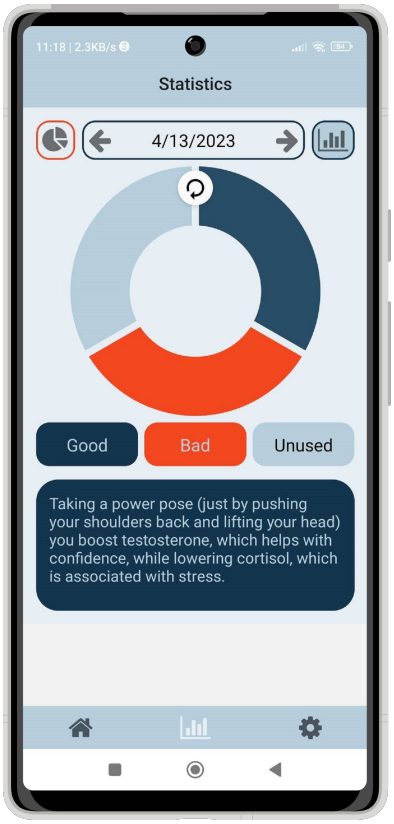
Értesítések

Az ülés káros az egészségre még abban az esetben is ha helyes testtartással túl sokáig végzi az ember. Ezért bizonyos időközönként szükség van egy kis mozgásra is [14]. Annak érdekében, hogy ezt a felhasználó észben tartsa és ne különböző emlékeztetőket kelljen beállítson a telefonján, az alkalmazás értesítéseket küld. Három különböző értesítés érkezik a felhasználónak, melyek közül kettő az üléssel kapcsolatos, egy pedig a széken levő akkumulátorral.

Az első értesítés típus akkor jelenik meg, ha a felhasználó helytelnül ül a széken. Egy bizonyos idő eltelte után jelez a telefon, hogy változtatni kell az ülési testtartáson. A második értesítés akkor jelenik meg ha a szék túl hosszú ideig használatban van, amely az egészségre már káros lehet. Ez az idő személyre szabható az alkalmazás beállításaiban. A harmadik típus az akkumulátor töltöttségi szintjéhez köthető. Ez az értesítés akkor érkezik, amikor az akkumulátor töltöttségi szintje alacsony (kevesebb mint 15%).

Pull to refresh

Az alkalmazásban szereplő diagramokat a székbe szerelt mikrovezérlőtől lekért adatok alapján lehet frissíteni. Mivel a népszerű alkalmazások körében jól bevállt módszer, hogy



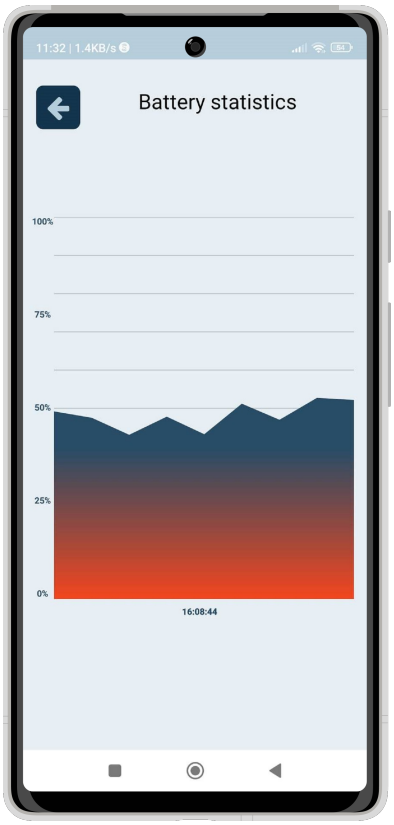
6. ábra. Pull to refresh

a képernyőn való lefele húzással frissül a képernyő tartalma, ezért az alkalmazásba is ez a módszer van beépítve az adatok szinkronizálására, a diagramok frissítésére. Ezt a 6. ábrán lehet megtekinteni.

Akkumulátor statisztika

A székre szerelt mikrokontrollert egy monitorizált akkumulátor modul látja el árammal. A töltöttségi szint számszerűen megjelenik a főnézet bal felső sarkában, egy akkumulátor ikonban. Ha töltés alatt áll az akkumulátor, akkor a számokat egy töltés ikon váltja fel. Az akkumulátor ikon megérintése után egy oszlopdiagramm jelenik meg az akkumulátor töltöttségi szintjének a változásáról.

A diagramon az eltelt idő és az akkumulátor töltöttségi szintjének a viszonylata látszik. A diagramot két szín alkotja: narancssárga, ha az akkumulátor töltöttségi szintje 15% alá csökken, illetve sötétkék ha fölötte van.



7. ábra. Akkumulátor statisztika

Beállítások

Az alkalmazást a felhasználó testreszabhatja különböző alapbeállítások segítségével. Ezek a beállítások a képernyő alján levő beállítás ikont érintve érhetőek el (8a. és a 8b. ábra).

Az alkalmazás ezen részén megtalálható, hogy a telefon melyik székhez van csatlakoztatva illetve a szék egyedi azonosítója is látható. A szék nevét megérintve a felhasználó megváltoztathatja ezt egy általa kiválasztott névre.

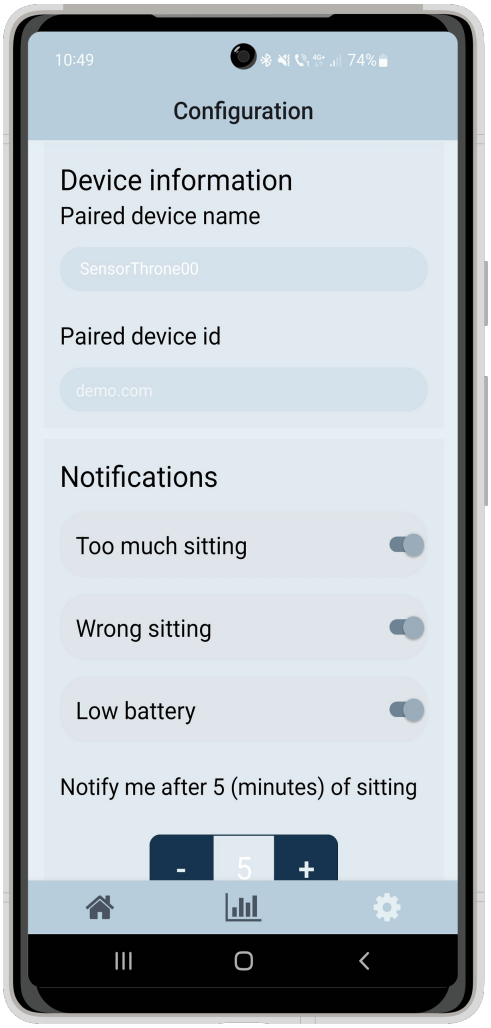
Lehetőség van beállítani, hogy a felhasználó miről szeretne értesítéseket kapni és milyen időközönként, erről bővebben a 2.2.1 fejezet alatt lehet olvasni.

A kijelző legalján látható a széknek, illetve az applikációnak a verzió száma. Ugyan itt van lehetőség bontani a kapcsolatot az előzőleg csatlakoztatott székkel.

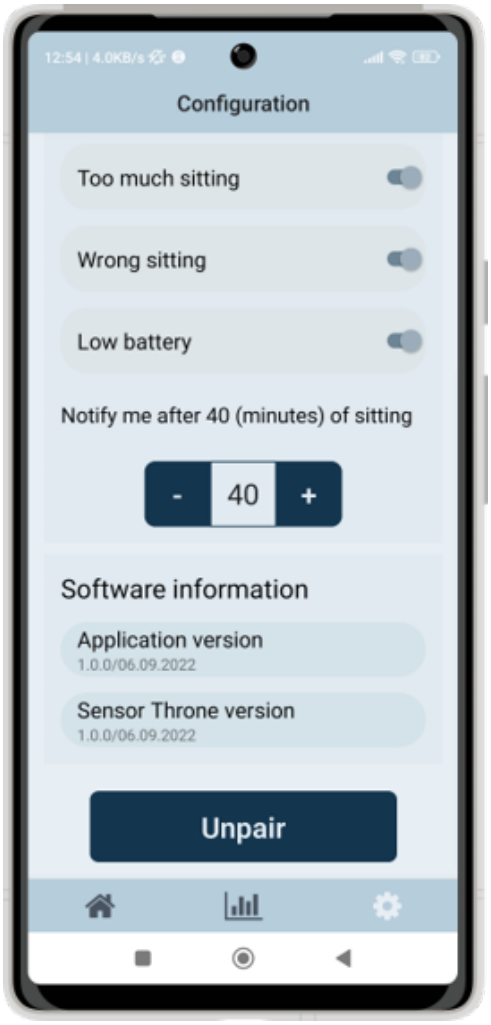
Több felhasználó támogatása

Az alkalmazás és a hardware komponens is támogat több felhasználót. Ez azt jelenti, hogy egy alkalmazás csatlakozhat több székhez, illetve egy szék csatlakozhat több alkalmazáshoz (nem egyidejűleg).

Az alkalmazás indulásakor automatikusan lementődik a felhasználó azonosítója a telefon



(a) Beállítások



(b) Beállítások

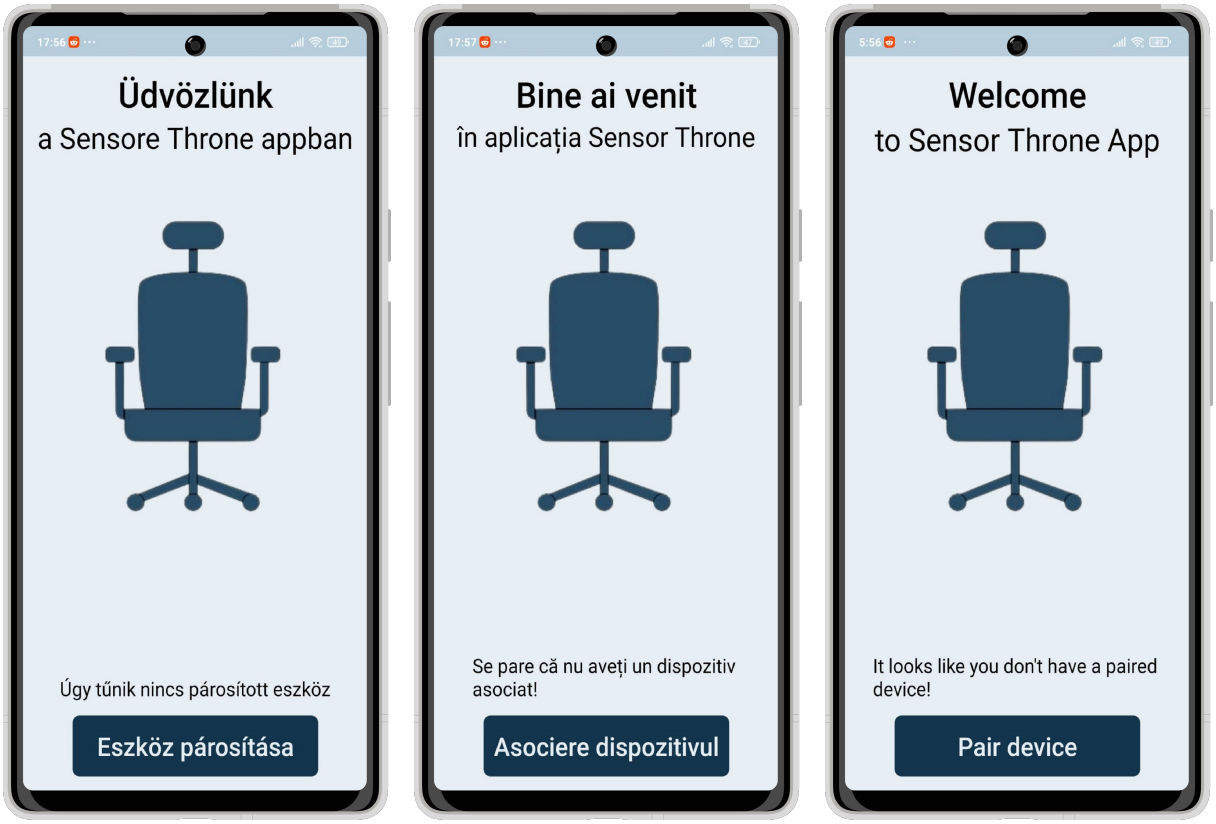
memóriájába, amely alapján a széken levő rendszer be tudja azonosítani a felhasználót. Emellett fontos szerepet játszik a kommunikációban, ahol az azonosító része a protokollnak. Erről bővebb leírás található az 2.3 rész alatt.

Nemzetköziesítés

Jelenleg az alkalmazás három nyelven érthető el: magyar, román és angol nyelven. A nyelvet a rendszer automatikusan választja ki a telefon nyelve alapján. A 9a., 9b. és a 9c. ábrákon a kezdőképernyő látható a fent említett három nyelven.

2.2.2. Nehézségek az engedélykérésekkel

Ahhoz, hogy a telefon Bluetooth modulját használni tudja az alkalmazás, különböző engedélyekre van szükség. Kezdetben elég volt egyetlen engedélyt kérni, mely a helymeghatározáshoz adott hozzáférést. Ez az engedély az `ACCESS_FINE_LOCATION` nevű



(a) Magyar nyelv (b) Román nyelv (c) Angol nyelv

9. ábra. Üdvözlő nézet három nyelven

engedély volt.

Az Android 12 rendszer támogatása problémát okozott a Bluetooth kapcsolat létrehozásakor, mivel oda már egyetlen engedély nem volt elég. Ugyanis az Andorid 12 rendszernél már nem elég a helyhez való hozzáférési engedély, hanem másik két Bluetooth specifikus engedélyre is szükség van. Egyik engedély a `BLUETOOTH_SCAN` melynek segítségével az elérhető eszközöket lehet megkeresni. Ahhoz, hogy a kommunikáció létrejöjjön a két Bluetooth eszköz között a `BLUETOOTH_CONNECT` engedély kell [3]. A működéshez az is elengedhetetlen, hogy a helymeghatározás is be legyen kapcsolva a telefonon, ezért erre is megoldást kellett keresni. A felhasználó engedélyével a helymeghatározás automatikusan bekapcsolódik.

Demo mód

Az alkalmazás fejlesztési ideje alatt létre lett hozva egy demo mód is a többi funkcionalitás mellett. Ezzel a funkcionalitással a fejlesztés is gördülékenyebben ment. A demo mód segítségével az alkalmazás összes funkcionalitását le lehet tesztelni a hardware

eszközök használata nélkül. Random generált statisztikákat lehet megtekinteni, a valós idejű monitorizálás is megjelenik, illetve a beállításokat is lehet kezelni.

2.3. Kommunikáció az alkalmazás és a szék között

Az alkalmazás és a székre felszerelt mikrokontroller között Bluetooth kapcsolaton keresztül történik a kommunikáció. A hardware és a software komponensek között zajló kommunikáció nyomonkövetésének és rendszerezésének érdekében egy kommunikációs protokoll bevezetésére volt szükség. A protokoll használata stabilabbá és biztonságosabbá teszi a rendszert, illetve hibakezelésre is lehetőséget ad.

```
APP goldfish_x86 0 REQ ONBOARDING step_s0~
PICO goldfish_x86 0 RESP ONBOARDING 0~
```

A fenti ábrán látható egy üzenetváltás az alkalmazás és a szék között. A protokoll hét részből áll, ezek egy-egy szóközzel vannak elválasztva, illetve egy $\sim$ karakterrel zárva:

1. A protokollal fomázott üzenet első tagját, az üzenetet előállító egység típus azonosítója alkotja, amely a következő értékeket veheti fel: `APP`, `PICO`. Ez mutatja, hogy az üzenet az applikáció, vagy a mikrokontroller oldaláról érkezik.

2. A protokoll második tagja az üzenetet küldő egység egyedi azonosítója, így pontosan tudható melyik telefonnal zajlik a kommunkáció.

3. A harmadik komponens a `Session ID`, amely egy szám értéket tartalmaz. Ez a számérték minden üzenetpár során eggyel növekszik. Az ID segítségével nyomonkövethető minden üzenet célbaérése, továbbá segít a hibakezelésben. Ha az egyik eszköz elküldött egy kérést és nem kapja meg a választ, akkor újraküldi az üzenetet, ezzel kiszűrve az adatvesztést és növelve a stabilitást.

4. A negyedik rész a parancs típusa, amely a következő értékeket veheti fel: `REQ`, `RESP`, `ACK`, `INFO` és `ERROR`. A protokoll meghatározása szerint egy `REQ` (kérés) üzenetre mindig egy `RESP` (válasz) üzenet kell érkezzen. Egy információt közlő üzenetben az `INFO` kulcsszó jelenik meg. Ha egy olyan üzenet sorozat következik, amiben egy parancson belül több üzenet váltás történik, akkor mindig az `ACK` kulcsszó használatával a vevő egy nyugtával hozza a küldő tudomására az üzenet megérkezését. Hiba esetén az `ERROR` parancs kerül be az üzenetbe.

5. Az üzenet ötödik komponense a parancs, ami megadja, hogy pontosan milyen művelet hajtódjón végre a mikrokontroller és az applikáció között. Ez a következő lehet: `ONBOARDING`, `GET_STATISTICS`, `SEND_SENSOR_STATUS`, `STOP_SENSOR_STATUS`, `SEND_PHONE_ID`, `GET_BATTERY`. `Onboarding` esetén az 2.2.1 fejezet alatt leírt folyamat kerül végrehajtásra. A `GET_STATISTICS` során szinkronizálás történik a szék és az alkalmazás között. A főképernyőre lépés esetén, a valós idejű monitorizálást a `SEND_SENSOR_STATUS` parancs teszi aktívvá, ha pedig más nézet van fókuszban, akkor a `STOP_SENSOR_STATUS` kulcsszó használandó. Abban az esetben, mikor a felhasználó csatlakozni szeretne a telefonjával a székhez, az alkalmazás automatikusan elküldi a mobil azonosítóját, ami a `SEND_PHONE_ID` parancs által történik meg. Az akkumulátor töltöttségi szintje a `GET_BATTERY` kód megadásával kérhető le.

6. Az utolsó előtti része az üzenetnek maga az információ, amit a küldő szeretne továbbítani. Ez egy szöveges karakterlánc, ami tömören tartalmazza az információt.

7. Az utolsó része az üzenetnek a zárókarakter, ami a protokoll alapján a $\sim$ karakter. Ezzel jelzi a küldő fél az üzenete végét.

3. Felhasznált technológiák

Ebben a fejezetben a projekt fejlesztése alatt felhasznált technológiák lesznek bemutatva. Ezek név szerint a következők: MicroPython, React Native és TypeScript.

3.1. MicroPython

A MicroPython egy karcsú és hatékony implementációja a Python 3 programozási nyelvnek, amely a Python standard könyvtárának egy kis részhalmazát tartalmazza és arra van optimalizálva, hogy mikrokontrollereken és korlátozott környezetben fusson [15].

A MicroPython kiválóan alkalmas az olyan projektekhez, amelyekben fontos az alacsony áramfogyasztás, a kis memóriaigény és az alacsony számítási teljesítmény. A projekt során a mikrokontroller szoftvere MicroPython nyelvben írodott, az előbb említett tulajdonságok miatt.

3.2. React Native

A React Native [17] egy Facebook által fejlesztett, nyílt forráskódú keretrendszer, amely a webes React keretrendszer és a natív fejlesztés kombinációja által teszi lehetővé mobilalkalmazások fejlesztését. Az alkalmazásokat általános JavaScript és React API-kra építi fel és lehetővé teszi a fejlesztést több platformra (Például: iOS és Android) egyszerre, jelentősen csökkentve a fejlesztési időt és költségeket. A fejlesztési idő csökkentését segíti elő az is, hogy amikor a kódban valami megváltozik nem kell az egész alkalmazást újra buildelni, hanem elég csak újratölteni azt.

Jelenleg a SensorThrone alkalmazás csak Android eszközökön érhető el, mivel napjainkban az legtöbb felhasználó eszközén Andoid található [1]. Hosszú távon tervben van az iOS platform támogatása is.

3.3. TypeScript

A TypeScript erősen típusozott programozási nyelv, amely a JavaScriptre épít, és jobb eszközöket nyújt a kódszervezéshez bármilyen méretű projektnél. A TypeScript további szintaxist ad a JavaScripthez, ezért korai hibafelismerés történik az editorban. A TypeScript a típusinferencia segítségével nagyszerű eszközöket nyújt kód írása során [22].

4. Fejlesztési eszközök és módszerek

A fejlesztési eszközök nagyban megkönnyítik és lerövidítik a fejlesztők dolgát. Ebben a fejezetben a SensorThrone projekten belül használt fejlesztési eszközökről és módszerekről lesz szó.

4.1. Scrum

A projekt fejlesztése alatt a csapat a Scrum [25] keretrendszert követte. Ez azt jelenti, hogy a fejlesztés két hetes periódusokra, úgynevezett sprintekre volt felosztva. A sprinteket egy tervezés előzte meg, melyben az adott periódusban megvalósítani kívánt funkcionalitások, feladatok lettek megbeszélve. Minden reggel rövid megbeszélések voltak tartva amelyen mindenki elmondta, hogy az azelőtti nap milyen feladatokat valósított meg, ütközött-e valamilyen nehézségbe, és milyen tervei voltak az adott napra. Ez nagyban elősegítette a csapatmunkát mivel a problémák esetén közösen lehetett megoldást találni. A sprint végén egy demo következett melyben bemutatásra kerültek a sprint alatt implementált funkcionalitások. Ezen részt vett a projekt ötlet gazdája illetve a fejlesztői csapat tagjai.

4.2. Git és Gitlab

A projekt fejlesztése alatt a forráskód változásainak a nyomonkövetése a Git [12] osztott verziókövető rendszer segítségével történt. A projekt számára két repository lett létrehozva: egy az alkalmazásnak és egy a mikrovezérlőt irányító szoftvernek. A létrehozott tárolók kezelése a Gitlab nevű webes szolgáltatáson keresztül történt. A projekt szétosztása illetve a két különböző repository megkönnyítette a párhuzamosan történő fejlesztést. Minden funkcionalitás fejlesztése külön ágon történt. Miután a funkcionalitás implementálása befejeződött, egy Merge request létrehozása segítségével a fejlesztői csapat átnézte és kijavította a kódot. Ezt követően az új funkcionalitás bekerült a már meglévők közé.

4.3. CI/CD Pipeline

A projekt fejlesztése során mindkét résznek létre lettek hozva a megfelelő pipelineok. Amikor módosítás történt a projekten, a Gitlab CI/CD [7] eszköze segítségével a pipelineok automatikusan lefutottak. Ezeknek köszönhetően a már meglévő kódrészlethez csak helyesen

```
stages:
  - check

flake8_test:
  stage: check
  image: python:3.9.13
  script:
    - pip install flake8
    - flake8
```

10. ábra. Hardware CI

szerkesztett kóddal bővült. Ezzel az eljárással kiszűrhetőek a fejlesztők által észre nem vett hibák.

A pipelineokat a `.gitlab-ci.yml` állomány segítségével lehet konfigurálni itt lehet beállítani, hogy milyen eszközök ellenőrizzék az új forráskódot és milyen szakaszai legyenek az ellenőrzésnek.

Feljebb látható a mikrovezérlőt irányító, micropythonban írt kódot ellenőrző pipeline. A kód csak egyetlen ellenőrzésen kell átmenjen mely a flake8 segítségével történik. A flake8 a Python programozási nyelv formai követelményeinek betartását segítő formázó.

Az alkalmazást ellenőrző pipeline (11. ábra) már két szakaszból áll amelyek sikeresen le kell fussanak ahoz, hogy a kódrészlet az elvárásoknak megfeleljen.

Az első fázis egy formai ellenőrzés majd ezt követi az alkalmazás buildelése. Ha mindkét fázis hiba nélkül lefut, akkor az új kódrészlet megfelel a követelményeknek.

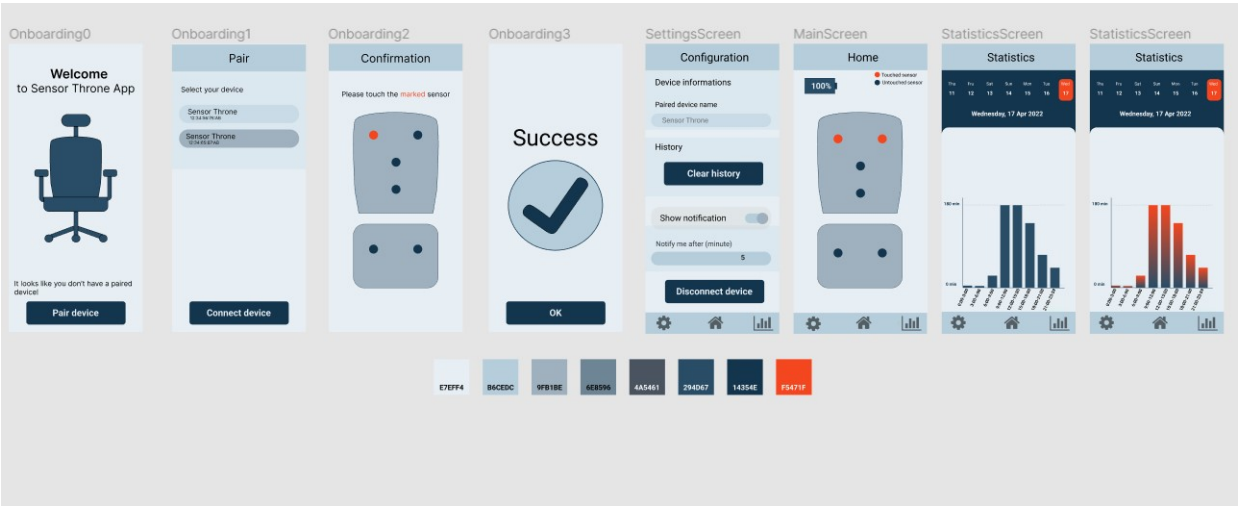
4.4. Figma

Annak érdekében, hogy az alkalmazás felhasználói felületének kinézete előre el legyen készítve egy tervezői program segítségére volt szükség. Egy ilyen felhasználói felületet tervező webalapú szoftver a Figma [11]. A szoftver lehetővé teszi a fejlesztők számára a webhelyek, mobilalkalmazások, ikonok kinézetének a megtervezését. Egyik előnye, hogy a benne létrehozott projekten több fejlesztő is dolgozhat egy időben, ami megkönnyíti és felgyorsítja a munkát és a tervezést.

A Sensor Throne alkalmazás felhasználói felületének a tervei ennek az eszköznek a segítségével készültek el (12. ábra). Itt volt megtervezve a különböző képernyőknek a kinézete melyek később megvalósításra kerültek.

```
image: reactnativecommunity/react-native-android

stages:
  - lint
  - build

before_script:
  - yarn install --frozen-lockfile

cache:
  paths:
    - node_modules

check:
  stage: lint
  script:
    - yarn lint
    - yarn tsc

build:
  stage: build
  script:
    - cd android  && ./gradlew assembleRelease
    - yarn install --pure-lockfile --cache-folder .yarn
```

11. ábra. Alkalmazás CI



12. ábra. Az alkalmazás felhasználói felületének terve

4.5. Thonny

A Thonny [20] egy Python programozási nyelvhez készített fejlesztői környezet. Könnyű telepíteni, illetve a felhasználói felülete is nagyon egyszerű. Mivel a MicroPythont támogatja, ezért ez volt használatban a mikrovezérlő beprogramozására. Egyik nagy előnye, hogy a segítségével könnyedén ki lehet telepíteni a mikrokontrollerre a megírt kódot. Ez az egyszerű kitelepítés több állomány esetén is működik.

5. Következtetések és továbbfejlesztési lehetőségek

A Sensor Throne nevű projekt keretén belül sikerült egy olyan széket és hozzá tartozó alkalmazást megvalósítani, mely a felhasználójának monitorizálja az ülési testtartását és különböző diagramokkal illetve értesítésekkel próbálja azt javítani.

A székre egy mikrovezérlő van szerelve, mely felelős a szenzorok megfigyeléséért illetve az adatok továbbításáért a telefon irányába. A kommunikáció Bluetoothon keresztül történik a mikrovezérlő és a telefon között.

A telefonos alkalmazás segítségével a felhasználó valós időben is láthatja ülési testtartását de a különböző diagramok alapján akár vissza is nézheti a statisztikákat. Ezen kívül beállításokhoz is hozzáfér a felhasználó, melyekkel az értesítéseket is testreszabhatja. Az alkalmazás emellett több nyelven is használható.

Az energiafogyasztás optimalizálása által az akkumulátor működési idejét lehetne jelentősen meghosszabbítani. Ezt az optimalizálást a „deep sleep" funkcionalitás bevezetésével lehet elérni a mikrovezérlő részén. Ezáltal ha a szék nincsen használatban, akkor a mikrovezérlő nem monitorizálja feleslegesen a szenzorok működését illetve egy olyan módba kerül amiben minimális mennyiségű energiát használ el és ezzel jelentős energiát lehet megspórolni.

Egy másik hasznos funckionalitás az Over the Air update és az automatikus deployment Firebase segítségével mely a Raspberry Pi Picon levő szoftver távoli frissítését tenné lehetővé. Ez előnyös lenne mivel így nem kellene minden elektronikai egységre külön rácsatlakozni a fejlesztőnek, hanem egy gombnyomásra a frissített szoftver települne a mikrovezérlőre.

A későbbiekben a felhasználók adatainak a gyűjtése is hasznos funkció lenne. Ezáltal akár következtetéseket is le lehetne vonni egy felhasználó szokásairól. Ezeket az adatokat akár az orvostudományban is fel lehetne haználni, illetve különböző kimutatásokat lehetne ezek alapján készíteni.

Hivatkozások

[1] *Android vs. Apple Market Share: Leading Mobile Operating Systems (OS) (Apr 2023)*. URL: https://www.bankmycell.com/blog/android-vs-apple-market-share/ (elérés dátuma: 2023. ápr. 30.)

[2] *Antares hivatalos oldala*. URL: https://scaune.ro/ (elérés dátuma: 2023. ápr. 29.)

[3] *Bluetooth permissions*. URL: https://developer.android.com/guide/topics/connectivity/bluetooth/permissions (elérés dátuma: 2023. ápr. 26.)

[4] Andrew Buck. *Mobile Apps vs Mobile Websites: Which is Best for 2023?* 2023. URL: https://www.mobiloud.com/blog/mobile-apps-vs-mobile-websites (elérés dátuma: 2023. ápr. 20.)

[5] Scott Campbelll. *BASICS OF THE I2C COMMUNICATION PROTOCOL*. URL: https://www.circuitbasics.com/basics-of-the-i2c-communication-protocol/ (elérés dátuma: 2023. ápr. 19.)

[6] CCOHS. *Working in a Sitting Position*. 2022. URL: https://www.ccohs.ca/oshanswers/ergonomics/sitting/sitting_overview.html (elérés dátuma: 2023. ápr. 19.)

[7] *CI/CD*. URL: https://about.gitlab.com/topics/ci-cd/ (elérés dátuma: 2023. ápr. 21.)

[8] *Codespring mentorprogram*. URL: https://edu.codespring.ro/codespring-nyari-szakmai-gyakorlat-2022/ (elérés dátuma: 2023. ápr. 29.)

[9] *Eredeti kép a Raspberry Pi Picoról*. URL: https://www.rpibolt.hu/img/15922/RPI-SC0918_altpic_1/RPI-SC0918.jpg?time=1679387877 (elérés dátuma: 2023. ápr. 30.)

[10] Mary Grace Legaspi Eric Peňa. *UART: A Hardware Communication Protocol Understanding Universal Asynchronous Receiver/Transmitter*. 2020. URL: https://www.analog.com/en/analog-dialogue/articles/uart-a-hardware-communication-protocol.html (elérés dátuma: 2023. ápr. 19.)

[11] *Figma hivatalos oldala*. URL: https://www.figma.com/ (elérés dátuma: 2023. ápr. 26.)

[12] *Git*. URL: https://git-scm.com/ (elérés dátuma: 2023. ápr. 21.)

[13] *Gregory Smart Chair*. URL: https://gregorychairs.com.au/ergonomics/smart/ (elérés dátuma: 2023. ápr. 20.)

[14] Madeline Holcombe. *Sitting too much is bad for your health, but offsetting the impact is easy, study shows*. 2023. URL: https://edition.cnn.com/2023/01/12/health/sitting-prolonged-study-wellness/index.html (elérés dátuma: 2023. ápr. 24.)

[15] *MicroPython hivatalos oldala*. URL: https://micropython.org/ (elérés dátuma: 2023. ápr. 20.)

[16] *Raspberry Pi Pico and Pico W*. URL: https://www.raspberrypi.com/documentation/microcontrollers/raspberry-pi-pico.html (elérés dátuma: 2023. ápr. 19.)

[17] *React Native hivatalos oldala*. URL: https://reactnative.dev/ (elérés dátuma: 2023. ápr. 20.)

[18] Nicole F. Roberts. *Americans Sit More Than Anytime In History And It's Literally Killing Us*. 2019. URL: https://www.forbes.com/sites/nicolefisher/2019/03/06/americans-sit-more-than-anytime-in-history-and-its-literally-killing-us/ (elérés dátuma: 2023. ápr. 19.)

[19] *Sit at work? You are one of 39%*. 2019. URL: https://ec.europa.eu/eurostat/web/products-eurostat-news/-/ddn-20190305-1 (elérés dátuma: 2023. ápr. 19.)

[20] *Thonny hivatalos oldala*. URL: https://thonny.org/ (elérés dátuma: 2023. ápr. 26.)

[21] *TTP223 Touch sensor*. 2020. URL: https://iotstarters.com/touch-based-switch-board-using-ttp223/ (elérés dátuma: 2023. ápr. 19.)

[22] *Typescript hivatalos oldala*. URL: https://www.typescriptlang.org/ (elérés dátuma: 2023. ápr. 20.)

[23] *UPS modul*. URL: https://www.waveshare.com/wiki/Pico-UPS-A (elérés dátuma: 2023. ápr. 26.)

[24] Elise Vogler. *Bluetooth vs. Wi-Fi Power Consumption*. URL: https://itstillworks.com/bluetooth-vs-wifi-power-consumption-17630.html (elérés dátuma: 2023. ápr. 24.)

[25] *What is scrum?* URL: https://www.scrum.org/learning-series/what-is-scrum (elérés dátuma: 2023. ápr. 21.)

[26] XUYUN ZENG. *Raspberry Pi Pico W Bluetooth firmware*. 2023. URL: https://www.makeuseof.com/ihow-to-try-bluetooth-on-the-raspberry-pi-pico-w/ (elérés dátuma: 2023. ápr. 19.)