UNIVERZITA SV. CYRILA A METODA V TRNAVE

FAKULTA PRÍRODNÝCH VIED

INTERAKCIA ČLOVEK-POČÍTAČ PRE INKLÚZIU: MOBILNÁ APLIKÁCIA PRE HRU NA KLAVÍR

Diplomová práca

2024 **Bc. Peter Hafner**

UNIVERZITA SV. CYRILA A METODA V TRNAVE
FAKULTA PRÍRODNÝCH VIED

INTERAKCIA ČLOVEK-POČÍTAČ PRE INKLÚZIU: MOBILNÁ APLIKÁCIA PRE HRU NA KLAVÍR

Diplomová práca

Študijný program: aplikovaná informatika
Študijný odbor: 18 – informatika
Školiace pracovisko: Ústav počítačových technológií a informatiky
Školiteľ: doc. Ing. Michal Čerňanský, PhD.

2024 **Bc. Peter Hafner**

UCM Trnava
Fakulta prírodných vied

ZADANIE ZÁVEREČNEJ PRÁCE

Meno a priezvisko študenta:	Bc. Peter Hafner
Študijný program:	aplikovaná informatika (Jednoodborové štúdium, magisterský II. st., denná forma)
Študijný odbor:	18. - informatika
Typ záverečnej práce:	Diplomová práca
Jazyk záverečnej práce:	slovenský
Sekundárny jazyk:	anglický

Názov: Interakcia človek-počítač pre inklúziu: mobilná aplikácia pre hru na klavír

Anotácia: 1. Analyzujte potreby a možnosti ľudí so zdravotným postihnutím pri hre na klavír, analyzujte možnosti pre zlepšenie prístupnosti prostredníctvom mobilnej aplikácie.
2. Analyzujte a overte vhodné technológie použiteľné pri tvorbe aplikácie pre podporu prístupnosti pri hre na klavír.
3. Navrhnite a implementujte aplikáciu, využite zvolené technológie.
4. Vytvorené riešenie overte a zhodnoťte.

Vedúci:	doc. Ing. Michal Čerňanský, PhD.
Katedra:	KAI - Katedra aplikovanej informatiky
Vedúci katedry:	PaedDr. Mgr. Miroslav Ölvecký, PhD. doc. RNDr. PaedDr. Ladislav Huraj, PhD.

Spôsob sprístupnenia elektronickej verzie práce:
bez obmedzenia

Dátum zadania: 20.10.2022

Dátum schválenia: 24.10.2022

vedúci katedry

Čestné vyhlásenie

Čestne vyhlasujem, že celú diplomovú prácu na tému „Interakcia človek-počítač pre inklúziu: mobilná aplikácia pre hru na klavír“ som vypracoval samostatne a pod odborným vedením školiteľa, s použitím uvedenej literatúry.

.....

Podpis študenta

Poďakovanie

Predovšetkým by som sa veľmi rád poďakoval môjmu školiteľovi doc. Ing. Michalovi Čerňanskému, PhD. za trpezlivosť, cenné a odborné rady pri vypracovávaní diplomovej práce.

Abstrakt

HAFNER, Peter: *Interakcia človek-počítač pre inklúziu: mobilná aplikácia pre hru na klavír*. [Diplomová práca]. Univerzita Sv. Cyrila a Metoda v Trnave, Fakulta prírodných vied, Ústav počítačových technológií a informatiky. Školiteľ: doc. Ing. Michal Čerňanský, PhD. Trnava, 2024. Počet strán: 71 s.

Práca popisuje tvorbu prístupného riešenia alternatívneho sustain pedálu, ktorý sa používa pri hre na klavír. Z riešenia môžu najviac ťažiť ľudia s postihnutím dolných končatín. Práca obsahuje analýzu súčasných riešení, návrh ľudskej interakcie s počítačom a jej implementáciu pre platformu iOS, ktorá implementuje rozšírenú realitu pre rozpoznávanie tváre a taktiež rôzne komunikácie medzi aplikáciou a klavírom. Hlavné časti aplikácie venujúce sa komunikácii implementujú MIDI sieťovú komunikáciu, Bluetooth Low Energy komunikáciu a hosťovanie hudobného nástroja inej aplikácie z AppStore, ktorý je zapojený do audio grafu našej aplikácie. Aplikácia je publikovaná v AppStore a je bezplatne dostupná po celom svete.

Kľúčové slová: Swift, ARKit, MIDI, BLE, AUv3, interakcia človek-počítač, prístupnosť.

Abstract

HAFNER, Peter: *Human-Computer Interaction for Inclusion: A Mobile Application for Piano Playing.*. [Master thesis]. University of St. Cyril and Methodius in Trnava, Faculty of Natural Sciences, Institute of Computer Technologies and Informatics. Supervisor: doc. Ing. Michal Čerňanský, PhD. Trnava, 2024. 71 p.

The work describes the creation of an accessible alternative sustain pedal solution used in piano playing. This solution can be most beneficial for individuals with lower limb disabilities. The work includes an analysis of current solutions, the design of human-computer interaction, and its implementation for the iOS platform, which incorporates augmented reality for face recognition and various communications between the application and the piano. The main parts of the application dedicated to communication implement MIDI network communication, Bluetooth Low Energy communication, and hosting a musical instrument from another application in the AppStore, which is integrated into the audio graph of our application. The application is published in the AppStore and is freely available worldwide.

Key words: Swift, ARKit, MIDI, BLE, AUv3, human-computer interaction, accessibility.

Obsah

Zoznam obrázkov

Zoznam tabuliek

Zoznam grafov

Zoznam skratiek

AUv3 – Audio Unit version 3

DAW – Digital Audio Workstation

MIDI – Musical Instrument Digital Interface

MVP – Minimal Viable Product

BLE – Bluetooth Low Energy

Úvod

Prístupnosť technológie pre používateľov so zdravotným postihnutím je dôležitou súčasťou informatiky, ktorá pomáha začleniť jednotlivcov do každodenného života. Nás zaujalo využitie technológie pre sprístupnenie kreatívnej činnosti – hre na klavír.

Ľudia s postihnutím dolných končatín majú problém s ovládaním klavírneho pedálu. Klasický sustain pedál sa nachádza na podlahe. Jeho funkcionalita spočíva v doznení nôt, ktoré sú hrané počas aktivovaného pedálu. Technika hrania s pedálom umožňuje vytvorenie zvuku, ktorý je bez pedálu nemožné vytvoriť.

Na internete sa nachádza niekoľko alternatívnych riešení, avšak žiadne z nich nie je určené pre širokú verejnosť. Sú to mechanické riešenia, ktoré by si musel každý skonštruovať sám, keďže sériová výroba z dôvodu nízkeho počtu potencionálnych zákazníkov roztrieštených po celom svete nepripadá do úvahy.

Riešenie v podobe mobilnej aplikácie sa javí ako najvhodnejší spôsob, ako dostať riešenie k čo najviac používateľom. Spoločnosť Apple vytvára produkty s ohľadom na ľudí so zdravotným postihnutím tým, že vytvára funkcie prístupnosti v systéme, ale umožňuje aj integráciu prístupnosti pre vývojárov aplikácií. Prístupnosť je jedna z hlavných hodnôt platformy Apple a preto sme sa okrem iného rozhodli vyvinúť natívnu aplikáciu pre iOS a iPadOS s názvom MouthPedal.

1 Analýza

1.1 Interakcia človek-počítač

Interakcia človek-počítač (HCI) je oblasť, ktorá sa zaoberá štúdiom, návrhom a implementáciou spôsobov, akými ľudia komunikujú s počítačmi a inými digitálnymi zariadeniami. Je to multidisciplinárny obor, ktorý kombinuje prvky informatiky, dizajnu, psychológie a ergonómie [1].

1.1.1 Historický kontext

Spotrebný počítačový priemysel začal poklesom ceny technológie, vďaka čomu sa k počítačom dostali aj iní ľudia, než tí z vlády, biznisu, či výskumu. Medzi prvými boli samozrejme nadšenci elektrotechniky, medzi ktorých patril aj Steve Paul Jobs, ktorý spolu so Stevenom Wozniakom zostrojili jeden z prvých domácich osobných počítačov Apple 1. V tejto kapitole popisujeme históriu interakcie človek-počítač využívanú v domácich osobných počítačoch na základe dostupných zdrojov [2].

Apple I – 1976

Hoci počítač na obrázku 1 nebol úplne prvým osobným počítačom, tak bol prvý, ktorý mal kompletnú základovú dosku a používateľ si k nemu pripojil svoje periféria ako klávesnicu a monitor. Tento počítač nepochybne vytvoril základ ľudskej interakcie s počítačom pre vývoj osobných počítačov.

Obrázok 1 – Apple I [3]

Apple II – 1977

O rok neskôr Apple predstavil svoj druhý počítač (obrázok 2), ktorý bol jedným z prvých počítačov, ktorý bol predávaný ako kompletný balík v krabici so všetkým potrebným na okamžité použitie. To znamená, že obsahoval základnú jednotku, ktorej súčasťou už bola vstavaná klávesnica. K jednotke sa externe pripájal monitor. Tento prístup urobil z Apple II jednoduchší a prístupnejší počítač pre koncových používateľov, keďže už nebol potrebný žiadny ďalší nákup periférií, alebo montáž počítača. Apple II tak bol jedným z prvých počítačov, ktorý bol navrhnutý s cieľom byť skutočne priateľský k používateľovi a ľahko dostupný pre domáce použitie. Najviac sa však rozmohol vďaka kancelárskej aplikácii VisiCalc. Takúto aplikáciu môžeme označiť ako "killer app". Teda aplikácia, kvôli ktorej by sme si kúpili daný počítač, hoci je určený pre všeobecné použitie. Na vzniknutí takéhoto konceptu sa priamo podieľa nový pohľad na ľudskú interakciu s počítačom.

Obrázok 2 – Apple II [4]

Xerox Alto a Macintosh - 1984

V 70. rokoch bol vyvinutý v Xerox Palo Alto Research Center (PARC) počítač Xerox Alto. Novinkou bol nový mechanizmus, ktorým počítač ovládame dodnes. Grafické používateľské rozhranie ovládané za pomoci myši skĺbovalo viacero technológií, ktoré umožnili túto interakciu s počítačom. Významným prínosom boli taktiež okná, vďaka ktorým bol používateľom umožnený multitasking. Boli taktiež prvý krát použité ikony, ktoré symbolicky reprezentovali akcie počítača. Tento prístup zaujal Steva Jobsa pri jeho návšteve v Xeroxe natoľko, že v roku 1983 Apple vydal počítač LISA, ktorý sa však vďaka jeho vysokej cene nestal populárnym. O rok neskôr Apple predstavil Macintosh (obrázok 3) a

potvrdil, že navrhnutá interakcia je správna. Dôležitým aspektom produktu bola teda jeho cena, ktorá pri predstavení počítaču Macintosh klesla z 10 000$ na 2 500$.

Obrázok 3 – Macintosh [5]

PowerBook až iBook

Jeden z prvých úspešných a ikonických prenosných počítačov PowerBook 100 (obrázok 4) navrhol James Buchanan. Pre ovládanie kurzora využíval trackball, ktorého nástupcom sa stal trackpad v PowerBooku 500. Steve Jobs v roku 1999 ukázal svetu prvý iBook prenosný počítač, ktorý sa dokázal pomocou WiFi pripojiť k sieti a tým ponúkol plne prenosný produkt v podobe, ako ho poznáme dnes.

Obrázok 4 – PowerBook 100 [6]

iPhone a iPad

Od predstavenia iBooku technológia umožnila vytvorenie výkonnejších a efektívnejších počítačov. Miniaturizácia prešla do úrovne, kedy v roku 2007 Steve Jobs predstavil svetu prvý skutočný dotykový smartfón (obrázok 5). Ovládanie pomocou prstu a gest bolo v produkte nevídaným prvkom. Z diskusií, ktoré po rokoch od predstavenia odhalili podrobnosti z vývoja sme sa dozvedeli, že Apple najprv pracoval na tablete iPad, ktorý chceli dodávať práve s dotykovým displejom. Keď však testovali prototyp, tak si rýchlo uvedomili, že takáto interakcia by mohla byť vhodná pre ovládanie telefónu. Spoločnosť preto predstavila iPad až neskôr v roku 2010. V rozhovoroch odhalili aj ako prebiehala práca na softvérovej klávesnici. Popisovali algoritmus, ktorý odhadoval, ktoré písmeno bude nasledovať a podľa toho upravoval veľkosť neviditeľnej plochy tlačidiel, pretože inak by sa používateľ nemusel trafiť, čo by spôsobovalo frustráciu. Prvý krát boli využité gestá, ako potiahnutie prstom v liste pre skrolovanie, zovretie dvoch prstov pre zmenšenie, či zväčšenie obsahu. Boli to nové interakcie, ktoré sú využívané dnes v každom mobilnom zariadení, čo svedčí o opodstatnenosti a kvalite ich návrhu a implementácie v danej dobe.

Obrázok 5 –– iPhone 2G [7]

Apple Vision Pro

Apple už v roku 2017 uviedol knižnicu ARKit, ktorá umožnila vývojárom integráciu rozšírenej reality. Tento krok bol, ako sme sa až v súčasnosti dozvedeli prvým v ceste ku headsetu Apple Vision Pro (obrázok 6). Tieto okuliare využívajú knižnice pre prácu s rozšírenou realitou, ktoré sú identické s knižnicami pre iné Apple zariadenia, čím urýchľujú vývoj aplikácii. Z pohľadu ľudskej interakcie máme možnosť pozorovať zaujímavú

interakciu ovládania. Ovládanie využíva oči ako kurzor a gesto prstov ako signál potvrdenia voľby. Môžeme si všimnúť, že toto zariadenie je kvôli nákladným moderným technológiám nedostupné pre širokú verejnosť. Avšak z histórie už poznáme tento vzor, kedy sa dostupnosť produktu zvýši až po znížení ceny.

Obrázok 6 – Apple Vision Pro [8]

1.1.2 Ciele HCI

Zo stručného historického prehľadu sme si mohli všimnúť, že interakcia s počítačom hrala veľkú rolu v použiteľnosti počítačov a v ich rozšírení z výskumných zariadení do domácností až do vrecka každého z nás. Stanovenie cieľov hralo veľkú rolu vo formovaní a preto uvádzame hlavné všeobecné ciele odvetvia interakcie človek-počítač [9].

Pri navrhovaní užívateľských rozhraní je dôležité porozumieť faktorom, ktoré ovplyvňujú spôsob, akým ľudia používajú technológiu. Na fyzickej úrovni sa oblasť HCI zaoberá výberom najvhodnejších vstupných a výstupných zariadení pre konkrétnu aplikáciu alebo úlohu. To zahŕňa aj určenie najlepšieho typu interakcie, či už ide o priamu manipuláciu, použitie prirodzeného jazyka, ikony, alebo menu. Okrem toho je nevyhnutné vyvíjať a zdokonaľovať metódy, ktoré umožňujú efektívnu interakciu s počítačovými systémami.

Tento integrovaný prístup k HCI umožňuje vytvárať užívateľsky prívetivejšie a efektívnejšie rozhrania, čo zvyšuje celkovú použiteľnosť a výkonnosť technologických produktov a systémov. Medzi hlavné ciele HCI patrí:

Bezpečnosť

Cieľom bezpečnosti v oblasti interakcie človek-počítač je zabezpečiť ochranu používateľa pred nežiadúcimi a potenciálne nebezpečnými situáciami. To znamená, že pri interakcii nesmie nastať riziko fyzického ublíženia na zdraví používateľa, či škoda na zariadení. Pri vykonávaní citlivých operácií, ako je vymazanie dát, je nevyhnutné zaviesť dodatočný

dialóg, ktorý overí zámer používateľa, aby sa minimalizovalo riziko neúmyselného, alebo neoprávneného vymazania dôležitých údajov.

Užitočnosť

Produkt by mal ponúkať primeraný rozsah funkcií a možností, ktoré sú potrebné na úspešné splnenie úlohy, ktorú používateľ musí, alebo chce vykonať. Vývojári by mali zabezpečiť, aby produkt efektívne zodpovedal potrebám používateľa a umožňoval im dosiahnuť ich ciele.

Účinnosť

Používateľ je schopný pomocou produktu splniť zadanú úlohu.

Efektivita

Cieľom je samozrejme optimalizácia času, za ktorý používateľ dokáže splniť danú úlohu.

Použiteľnosť

Použiteľnosť je dôležitým aspektom HCI, ktorý zabezpečuje, že produkt je jednoducho ovládateľný a použiteľný pre používateľa bez potreby štúdia rozsiahlych príručiek, alebo tréningu. Jeho používanie by malo byť intuitívne a ľahko naučiteľné, čo znamená, že používatelia by mali byť schopní rýchlo pochopiť, ako produkt funguje a efektívne ho používať bez zbytočnej frustrácie.

Dojem

Cieľom je poskytnúť používateľovi pozitívny dojem nielen po prvom kontakte s produktom, ale aj po dlhšom používaní. Produkty by mali udržiavať svoju použiteľnosť a atraktivitu, aby používatelia zostali spokojní a lojálni dlhodobo. V podobe softvéru to môže zahŕňať aj aplikovanie aktuálneho dizajnového trendu.

1.1.3 Definovanie konkrétnej interakcie

Špecifikácia interakcie je dôležitá pre jej pochopenie a následnú implementáciu. Uvádzame príklad definovania konkrétnej interakcie ako je to bežné v odvetví HCI [9]:

Príklad**: Lampa**

Funkcionalita: Osvetlenie miestnosti

Rozhranie: Tlačidlo pre zapnutie

Funkčná časť: Žiarovka

Interakcia: Stlačenie tlačidla - zapnutie svetla

Úloha používateľa: Zapnúť svetlo

1.1.4 Benefity HCI

Medzi najdôležitejšie výhody kvalitného návrhu a implementácie princípov HCI patria:

Získanie podielu na trhu

HCI, teda interakcia človek-počítač, hrá kľúčovú úlohu pri získavaní trhového podielu pre rôzne produkty a služby. Používatelia majú tendenciu vyberať produkty, ktoré sú ľahko ovládateľné a poskytujú vysokú úroveň použiteľnosti. Príkladom je vyhľadávací nástroj Google, ktorý si získal dominantnú pozíciu na trhu vďaka svojej jednoduchosti a efektívnosti. Podobne, zariadenia ako iPhone od spoločnosti Apple získali popularitu vďaka ich priateľskému užívateľskému rozhraniu a schopnosti poskytnúť používateľom pohodlný zážitok.

Zlepšenie produktivity

HCI prispieva k zlepšeniu produktivity pracovníkov a organizácií tým, že umožňuje rýchlejšie a efektívnejšie vykonávanie úloh. Napríklad, v spoločnosti, kde zamestnanci potrebovali dlhý rad kliknutí na vykonanie jednej úlohy, môže študent informatiky pomôcť vytvorením programu, ktorý automatizuje tento proces. Tým sa zvýši rýchlosť a efektivita práce, čo má za následok zvýšenú produktivitu a zlepšenie výkonu organizácie.

Zníženie nákladov na podporu

Vysoká úroveň použiteľnosti a efektívnosti produktov v oblasti HCI môže viesť k výraznému zníženiu nákladov na zákaznícku podporu. Ak je produkt ľahko ovládateľný a použiteľný, používatelia budú menej pravdepodobne potrebovať pomoc zákazníckej podpory. Napríklad, ak je použitie práčky náročné aj po prečítaní návodu, mnoho používateľov sa obráti na zákaznícku podporu, čo môže viesť k zvýšeným nákladom pre spoločnosť.

Zníženie nákladov na vývoj

Implementácia princípov HCI môže viesť k zníženiu nákladov na vývoj softvéru a technologických produktov. Identifikácia a odstránenie zbytočných funkcií, ktoré používatelia nechcú, môže znížiť čas a zdroje potrebné na vývoj a údržbu produktov. Napríklad, ak sú v textovom editore prítomné nadbytočné dialógové okná na potvrdenie, je pravdepodobné, že tento produkt bude vyžadovať prepracovanie, čo môže mať za následok zvýšené náklady na vývoj.

1.2 Metódy a metriky HCI

V tejto kapitole sme analyzovali metódy a metriky na základe dostupnej literatúry [10].

1.2.1 Všeobecné zásady HCI dizajnu

- Uspokojujúce
- Zábavné
- Príjemné
- Užitočné
- Prekvapujúce
- Esteticky príjemné
- Odmeňujúce
- Podporujúce kreativitu
- Emocionálne uspokojivé
- Nudné
- Frustrujúce
- Otravné

1.2.2 Normanove zásady použiteľnosti

Všeobecné zásady nám neprezrádzajú, ako ich priamo aplikovať. Z tohoto dôvodu boli vytvorené Normanove zásady použiteľnosti. Tieto zásady sa dajú priamo aplikovať a sú využiteľné aj v sférach, kde nie je nutne využitý počítač. Základné myšlienky Normanových zásad použiteľnosti sú [10]:

- možnosť učiť sa z bežných objektov

- možnosť ovládať bežné objekty bez požívateľského manuálu

- objekty by mali poskytovať nejaké náznaky ako ich používať

Normanové zásady použiteľnosti sú:

Zásada viditeľnosti

Užívateľ by mal jasne vedieť, ako funguje dané zariadenie a ako s ním interaguje na základe podnetov, ktoré pôsobia na jeho zmyslové orgány. Čím sú jeho funkcie jasnejšie, tým jednoduchšie je pre užívateľa určiť, čo má robiť ďalej.

Zásada spätnej väzby

Zariadenie by malo poskytovať užívateľovi spätnú väzbu, ktorá informuje o tom, aká konkrétna akcia bola vykonaná. Toto umožňuje užívateľovi pokračovať v činnosti a zabezpečuje lepšie porozumenie stavu systému.

Zásada obmedzení: Užívateľovi by mali byť obmedzené možné akcie, ktoré môže vykonať v danom okamihu. Tieto obmedzenia pomáhajú predchádzať nesprávnemu použitiu zariadenia a udržiavajú užívateľa na správnej ceste.

Zásada náznakov: Užívateľ by mal mať jasné náznaky o tom, ako používať dané zariadenie. Tieto náznaky súvisia s atribútmi objektu, ktoré naznačujú jeho funkcie a spôsob použitia.

Zásada mapovania: Existuje prirodzený vzťah medzi ovládacími prvkami a ich účinkami. Mapovanie zabezpečuje, že užívateľ je schopný intuitívne pochopiť, ako sa jeho akcie prejavujú v systéme.

1.2.3 System Usability Scale (SUS)

System Usability Scale (SUS) je spoľahlivý nástroj na meranie subjektívne vnímanej použiteľnosti produktov a služieb. Bol vyvinutý v 80. rokoch Johnom Brookom pre spoločnosť Digital Equipment Corporation s cieľom poskytnúť prostriedok na zistenie použiteľnosti počítačových systémov. Odvtedy sa SUS stal široko používanou metódou v oblasti testovania použiteľnosti a umožňuje hodnotiť rôzne typy produktov a služieb, vrátane hardvéru, softvéru, mobilných zariadení, webových stránok a aplikácií. SUS sa zvyčajne prezentuje vo forme dotazníka, ktorý pozostáva z desiatich položiek, pričom každá položka má päť možných odpovedí, čo umožňuje užívateľom ohodnotiť ich interakciu s produktom [11].

Otázky SUS dotazníka:

1. Produkt by som v prípade potreby rád/a použil/a.
2. Produkt je zbytočne komplexný a zložitý.
3. Produkt sa mi dobre používal.
4. Potreboval/a by som technickú podporu, aby som mohol/a produkt používať efektívne.
5. Rôzne funkcie produktu sú dobre spracované.
6. Produkt je nekonzistentný.
7. Povedal/a by som, že väčšina ľudí sa naučí s produktom pracovať rýchlo.
8. Produkt je ťažkopádny a ťažko sa ovláda.
9. Pri práci s produktom sa cítim pohodlne a isto.
10. Musím sa naučiť veľa vecí, aby som vedel/a s produktom pracovať.

Tieto otázky zodpovedajú rôznym aspektom použiteľnosti produktu a umožňujú získať komplexný pohľad na to, ako ho užívatelia vnímajú. Po vyplnení dotazníka sa odpovede skórujú a vypočítava sa celkové SUS skóre, ktoré poskytuje informácie o úrovni použiteľnosti produktu zo subjektívneho hľadiska používateľov.

Výpočet skóre:

Skóre SUS sa vypočítava z odpovedí na desať otázok SUS dotazníka, pričom každá odpoveď sa hodnotí na 5-bodovej Likertovej škále, kde:

- Rozhodne nesúhlasím: 1 bod

- Skôr nesúhlasím: 2 body

- Neviem: 3 body

- Skôr súhlasím: 4 body

- Rozhodne súhlasím: 5 bodov

Pre výpočet celkového SUS skóre sa vykonajú nasledujúce kroky:

Nepárne otázky:

Sčíta sa celkové skóre pre všetky odpovede na nepárne otázky a od tohto čísla sa odpočíta 5.

Párne otázky:

Spočíta sa celkové skóre pre všetky odpovede na párne otázky a toto číslo odpočítame od 25.

Finálne skóre:

Celkové skóre z krokov 1 a 2 sa spojí a vynásobí sa číslom 2,5, čím sa získa finálne SUS skóre vo forme percentuálnej hodnoty zo 100.

Interpretácia výsledkov:

SUS nie je diagnostický nástroj, čo znamená, že jeho cieľom nie je definovať konkrétne problémy, ale poskytnúť informácie, ktoré poukazujú na použiteľnosť produktu. Skóre nižšie ako 68 naznačuje pravdepodobné vážne problémy s použiteľnosťou produktu, ktoré je potrebné preskúmať a riešiť, zatiaľ čo skóre 68 a vyššie indikuje relatívne dobrú použiteľnosť produktu, pričom môže byť potrebné len niekoľko menších vylepšení. Pri interpretácii SUS skóre je tiež dôležité zvážiť individuálne odpovede na jednotlivé otázky a identifikovať oblasti, v ktorých je potrebné zlepšenie z pohľadu užívateľskej skúsenosti.

1.3 Prístupnosť

Prístupnosť v informatike [12] je oblasť, ktorá sa zaoberá dizajnom a vývojom informačných a komunikačných technológií (IKT) tak, aby ich mohli používať všetci ľudia, bez ohľadu na ich fyzické, zmyslové, kognitívne, alebo technické možnosti.

1.3.1 Dôležitosť prístupnosti

Prínos prístupných systémov a zariadení sa dotýka jednotlivcov, ale aj samotnej spoločnosti [12]:

Rovnosť príležitostí

Prístupnosť umožňuje ľuďom so zdravotným postihnutím plne participovať na spoločenskom živote, vrátane vzdelávania, zamestnania a prístupu k informáciám. Odhaduje sa, že na svete žije viac ako 1 miliarda ľudí so zdravotným postihnutím, z ktorých mnohí čelia prekážkam pri používaní IKT. Prístupný dizajn im umožňuje prekonať tieto prekážky a plne sa zapojiť do spoločnosti.

Ekonomický prínos

Prístupný trh je obrovský a predstavuje značný ekonomický potenciál pre firmy a organizácie. Štúdie ukazujú, že firmy, ktoré sa zameriavajú na prístupnosť, dosahujú lepšie výsledky a zvyšujú si zisky. Prístupné produkty a služby sú dostupné širšiemu okruhu zákazníkov, čo vedie k rastu a prosperite.

Etická zodpovednosť

Je našou etickou povinnosťou vyvíjať technológie, ktoré sú dostupné pre všetkých. Každý človek má právo na prístup k informáciám a službám bez ohľadu na svoje individuálne vlastnosti. Prístupný dizajn je prejavom inklúzie a rešpektu voči všetkým ľuďom.

1.3.2 Spoločnosť Apple a jej prístup

Prístupnosť patrí medzi hlavné hodnoty spoločnosti Apple. Funkcionalitám pre zdravotne postihnutých sa spoločnosť venuje od samotného začiatku platformy iOS. Funkcionality sú integrované v samotnom operačnom systéme, ale aj v aplikáciách vývojárov tretích strán, vďaka knižnici, ktorú Apple pripravil. To znamená, že pri každej novej funkcionalite kladú dôraz na sprístupnenie. To sa očakáva aj od vývojárov tretích strán, čomu svedčia vzdelávacie materiály, ktoré sú užitočné pri návrhu a implementácii rozhrania aplikácie. V nastaveniach systému sú funkcionality rozdelené do kategórií, na základe zdravotného postihnutia [13]:

Reč

Pokrok v strojovom učení umožnil vytvoriť funkcionalitu, ktorá dokáže vytvoriť syntetizovaný hlas, ktorý znie ako hlas konkrétneho človeka. Niektorí pacienti vedia, že čoskoro stratia svoj hlas a preto sa môžu rozhodnúť pre vytvorenie digitálnej podoby svojho hlasu do budúcnosti. Tento proces spočíva v nahratí 150 fráz a následnom trénovaní, ktoré prebieha priamo na zariadení cez noc, alebo iný vhodný moment. Po vytvorení je hlas možné využiť pri hlasových, či video hovoroch, alebo v bežných rozhovoroch. Hlasová asistentka Siri v špeciálnom režime čaká na potencionálnu komunikáciu dlhší čas a až potom začne na povel reagovať, čo umožňuje pohodlnejšiu interakciu pre určité zdravotné postihnutia, či úrazy.

Zrak

Pre nevidiacich ľudí slúži funkcionalita VoiceOver, ktorá predčíta obsah zvoleného prvku. V tomto režime je úplne nový pohľad na to, ako sa ovláda dotyková obrazovka. Používateľ ťuknutím len zvolí daný prvok na predčítanie. Pre potvrdenie akcie je nutný dvojklik. Pri potiahnutí prsta doprava a doľava sa postupne presúvame cez všetky prvky obrazovky. Pre posúvanie po obrazovke sa využíva gesto ťahu troch prstov v smere posúvania prvku obrazovky. Pre ľudí s obmedzeným videním sú dostupné nastavenia pre zmenu veľkosti textu, či zväčšenie celej obrazovky, pričom pohyb po nej je podobne zabezpečený gestom ťahu troch prstov. Pohyb prvkov na obrazovke môže viesť k rôznym zdravotným rizikám a preto je možné naprieč systémom obmedziť akékoľvek animácie, či vyhnúť sa v podporovanom médiu pasážam, ktoré sú označené ako blikajúci obraz.

Mobilita a motorika

Ľudia, ktorý majú problém s používaním dotykovej obrazovky, alebo potrebujú adaptívne príslušenstvo, využívajú funkcionalitu s názvom AssistiveTouch. Podporované príslušenstvo sa pripája pomocou technológie bluetooth. Toto príslušenstvo má štandardizované vstupné signály. Podporované sú aj populárne herné ovládače a aj ovládač pre Apple TV, čo môže byť často dostupnejšou alternatívou pre určité použitia. Okrem toho sa v nastaveniach nachádza množstvo možných prispôsobení interakcie s fyzickými, či softvérovými tlačidlami

Sluch

Operačný systém iOS podporuje načúvacie prístroje, ktoré sú plne prepojené s operačným systémom. Do načúvacieho prístroja sú prenášané všetky systémové zvuky, hudba, či audio

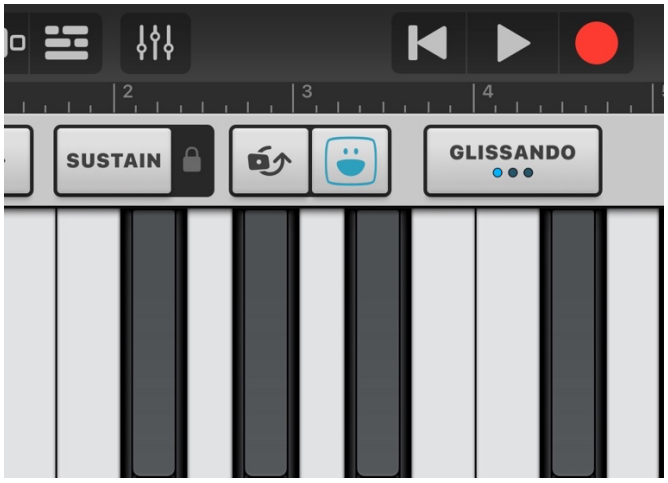
z videa. Mikrofón iPhonu môže byť použitý ako zdroj audia pre načúvací prístroj, ale taktiež aj pre slúchadlá AirPods. Pre nepočujúcich ľudí je taktiež dostupná funkcionalita, ktorá neustále pomocou mikrofónu sleduje okolie a dokáže používateľa upozorniť napríklad na požiar, tečúcu vodu, domáci zvonček, či zvuky zvierat. Priamo na zariadení používateľ dokáže natrénovať model pre rozpoznávanie zvuku vlastných domácich spotrebičov, či iné vlastné zvuky, na ktoré chce byť upozornený.

1.4 Hudobné aplikácie s podporou AR

Našim cieľom je pomôcť ľuďom s postihnutím pri hre na klavír pomocou AR a preto sme sa pozreli na hudobné aplikácie, v ktorých vývojári využívajú ARKit framework pre rôzne účely. Inšpirovať sa mohli priamo pri aplikácii zo štúdia Apple – GarageBand.

1.4.1 GarageBand

GarageBand pre iOS (obrázok 7) je hudobná aplikácia vyvinutá spoločnosťou Apple, ktorá umožňuje užívateľom tvoriť hudbu priamo na svojich mobilných zariadeniach. Táto aplikácia poskytuje širokú škálu nástrojov a funkcií na nahrávanie, úpravu a mixovanie hudobných skladieb. Jednou z unikátnych funkcií je Face Control, ktorý umožňuje používateľom interaktívne ovládať niektoré zvukové efekty pomocou pohybov tváre. Tento prvok technológie sledovania tváre otvára nové možnosti pre osobitý a zábavný spôsob tvorby hudby na mobilných zariadeniach. Aplikácia GarageBand je pravidelne aktualizovaná a jej inštalácia je bez poplatku [14].

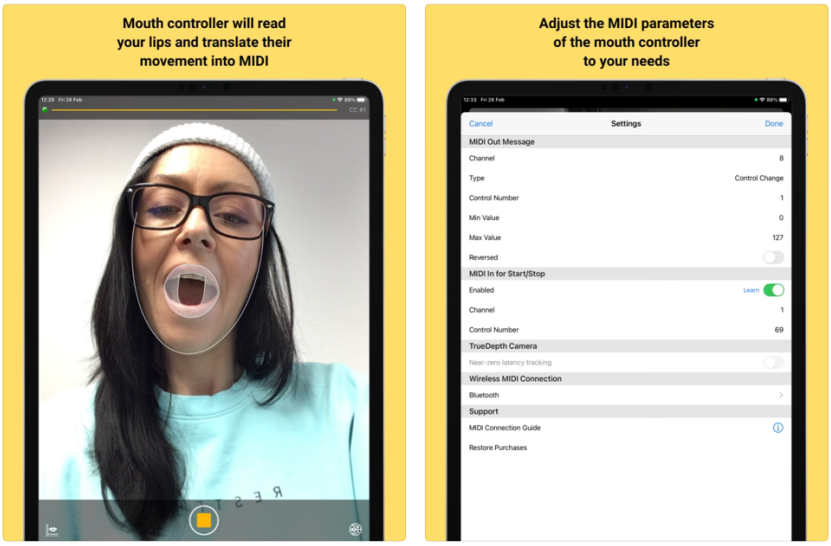


Obrázok 7 – GarageBand – Face Control

1.4.2 MIDI Mouth Controller

Podobnú funkcionalitu ponúka aj aplikácia MIDI Mouth Controller (obrázok 8), ktorá umožňuje túto funkcionalitu používať spolu so stolnou verziou GarageBand, či iným

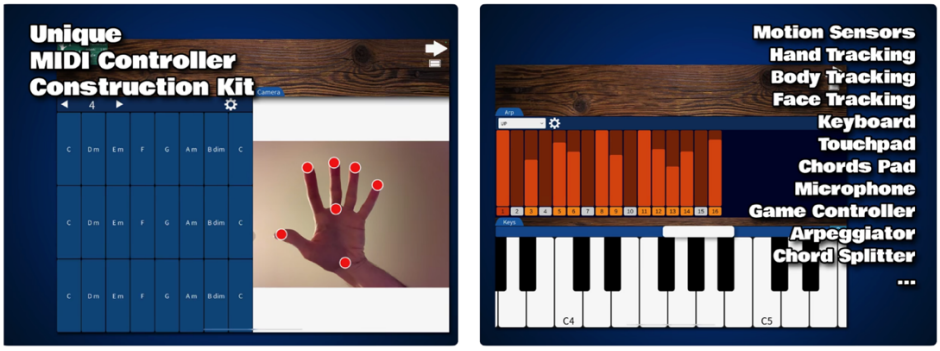
nahrávacím softvérom (DAW). Komunikácia je možná pomocou MIDI Network a MIDI over BLE. Aplikáciu sa nám nepodarilo nakonfigurovať pre pohodlné ovládanie MIDI sustain pedálu, pretože aktivačná hodnota pedálu je dosiahnutá až pri plnom otvorení úst. MouthController nebola aktualizovaná 3 roky a je možné ju zdarma vyskúšať, no jej plná verzia je za poplatok [15].



Obrázok 8 – MIDI Mouth Controller [15]

1.4.3 MusiKraken

MusiKraken (obrázok 9) je experimentálna hudobná aplikácia, ktorá využíva rôzne senzory pre ovládanie hudobných nástrojov. Medzi senzory patrí pohybový senzor, displej, mikrofón, ale aj kamera, ktorá je využitá pre AR rozpoznávanie častí tela, ktoré sa vďaka aplikácii stávajú ovládacími prvkami nástroja. Aplikácia získala ocenenie MIDI Inovation Awards. MusiKraken je platená aplikácia, ktorú však jej vývojár pravidelne aktualizuje. Hoci je aplikácia robustná a ponúka mnoho funkcionalít, nie je možné ju nakonfigurovať pre pohodlné ovládanie MIDI sustain pedálu. Aktivačná hodnota pedálu je dosiahnutá až pri plnom otvorení úst [16].



Obrázok 9 – MusiKraken [16]

1.5 Analýza riešeného problému

1.5.1 Formulácia riešeného problému

Ľudia s postihnutím dolných končatín majú problém s ovládaním klavírneho sustain pedálu. Pri súčasnom stave riešenia problematiky vidíme priestor na zlepšenie a preto je naším cieľom navrhnutie a implementovanie alternatívneho riešenia pre ovládanie klavírneho pedálu.

1.5.2 Súčasný stav

V súčasnosti exitujú viaceré alternatívne riešenia pre realizáciu efektu doznenia, avšak žiadne z nich nie je určené pre širokú verejnosť. Zvyčajne sú to mechanické riešenia, ktoré vyžadujú individuálnu konštrukciu, keďže sériová výroba z dôvodu nízkeho počtu potencionálnych zákazníkov by pravdepodobne nebola rentabilná. Riešenie v podobe mobilnej aplikácie sa javí ako najvhodnejší spôsob umožňujúci širokej skupine používateľov so zdravotným postihnutím obohatiť hru na klavír o efekt doznenia. Pred začatím samotného vývoja sme analyzovali súčasný stav a z dostupných zdrojov sme identifikovali viaceré rôzne prístupy, ktoré sa pri hre na klavír používajú [zdroje]. Prístupy sme porovnali spolu s naším riešením za pomoci kľúčových vlastností z hľadiska používateľského zážitku. Výsledok porovnania sa nachádza v tabuľke 1.

Druhá osoba (OSB, obrázok 10) – používateľ využíva asistenciu druhej osoby vzdelanej v klavírnej teórii, ktorá aktivuje, či deaktivuje pedál. Alternatívne neznalá osoba ovláda pedál na základe dohodnutého gesta klaviristu.

Obrázok 10 – Druhá osoba

Ústny spínač (UST, obrázok 11) – používateľ využíva tlak vyvinutý čelusťou na ústny spínač, ktorý aktivuje mechanizmus pedálu. Alternatívna možnosť je realizovať spínač, ktorý reaguje na fúknutie [17].

Helma so senzorom (HLM, obrázok 11) – používateľ má na hlave helmu, ktorá má v sebe zabudovaný pohybový senzor, ktorý pri naklonení hlavy aktivuje pedál [17].

Obrázok 22 – Helma so senzorom a ústny spínač[17]

Slúchadlový senzor (SLU, obrázok 13) – používateľ má v uchu slúchadlo, ktoré slúži aj na upevnenie senzora k hlave používateľa. Používateľ rovnako ako pri R3 náklonom hlavy aktivuje pedál [18].

Obrázok 33 – Slúchadlový senzor [18]

Dostupnosť – riešenie je jednoducho dostupné pre potencionálneho záujemcu.

Diskrétnosť – riešenie je nenápadné z pohľadu poslucháča. To môže dodávať sebavedomie vystupujúcemu (používateľovi).

Samostatnosť – riešenie nevyžaduje druhú osobu.

Voľný pohyb – riešenie umožňuje používateľovi voľný pohyb torza.

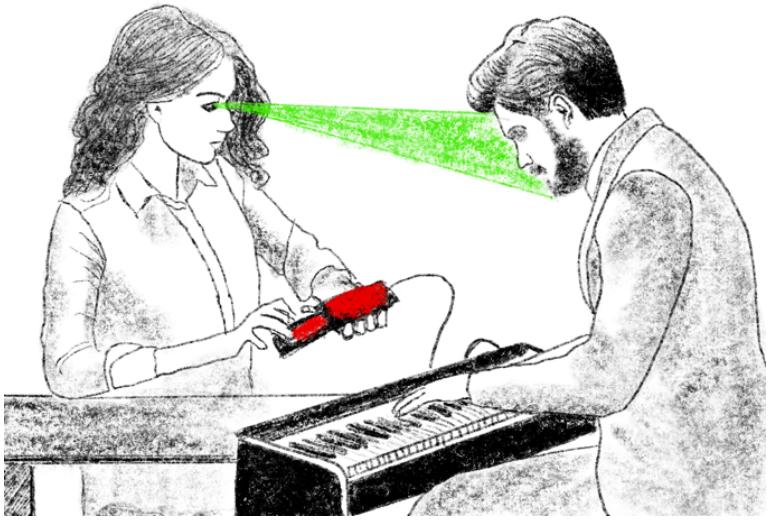
	OSB	UST	HLM	SLU	Mouth Pedal
Dostupnosť	*	*	*	*	*
Diskrétnosť	*	*	*	*	*
Samostatnosť	*	*	*	*	*
Voľný pohyb	*	*	*	*	*

Tabuľka 1 – Porovnanie riešení

2 Návrh a implementácia

2.1 Návrh interakcie

Identifikovanie kľúčových vlastností vhodného riešenia nám umožnilo navrhnúť nový spôsob interakcie, ktorý sme experimentálne overili. Základnou myšlienkou návrhu je umelé nahradenie druhej osoby z riešenia OSB počítačom. Naučiť počítač klavírnu teóriu a každú pieseň by bolo náročné a hlavne by nepodporilo kreativitu hudobníka. Preto sme potrebovali navrhnúť gesto, ktorým by používateľ dal počítaču signál pre aktiváciu pedálu, podobne ako používateľ druhej osobe v riešení OSB. Počítačové videnie dokáže rozpoznávať rôzne časti tela. Pri hre na klavír nepripadá do úvahy ovládanie pomocou rozpoznávania rúk. Ostáva nám teda iba rozpoznávanie tváre. Metódy pre detekciu tváre dokážu rozpoznávať jej rôzne samostatné časti. Ako rozpoznateľné gestá tváre, ktoré dokážu reprezentovať dva stavy (aktivovaný/deaktivovaný pedál), sme vo všeobecnosti identifikovali žmurknutie očami, dvihnutie obočia, otváranie úst, výraz úsmevu a nakoniec náklon hlavy smerom k zariadeniu. Pre otestovanie gest sme navrhli nenákladný experiment, ktorý spočíval v asistencii druhej osoby, ktorá sledovala gestá tváre počas hry používateľa a na ich základe aktivovala pedál. Obrázok 14 slúži pre vizualizáciu nenákladného prototypu interakcie.



Obrázok 44 – Nenákladný prototyp

Experiment ukázal, že ako najvhodnejšie riešenie sa javí použitie gesta otvorenia úst, samozrejme za predpokladu, že budeme schopní overiť možnosti technológie rozpoznať jemné pohyby úst. Náklon hlavy sa taktiež javí ako použiteľné riešenie, avšak nespĺňa

vlastnosť volaného pohybu pri hre. Ostatné gestá sa preukázateľne javia ako nevhodné. Porovnanie gest je možné vidieť v tabuľke 2.

Obočie	Po krátkom čase nepríjemné
Oči	Po krátkom čase nepríjemné a nevhodné, keďže v určitých pasážach skladby by musel mať používateľ zavreté oči – nevhodné pre začiatočníkov.
Ústa	Pri malom pootvorení úst interakcia neprekážala a ani po dlhšom čase hrania. Pôsobí prirodzene a nenápadne.
Úsmev	Nemusí sa zhodovať s emóciou hranej skladby.
Náklon hlavy	Ovplyvňuje prirodzený pohyb torza rovnako ako R3 a R4. Pri frekventovanej aktivácii mierne otravné, avšak použiteľné.

Tabuľka 2 – Porovnanie gest interakcie

2.1.1 Návrh interakcie s podporou Normanových zásad použiteľnosti

Po identifikácii správneho gesta sme pri návrhu aplikovali Normanove zásady použiteľnosti:

Zásada viditeľnosti

Dominantný prvok na obrazovke jasne reprezentuje stavy otvorených a zavretých úst prostredníctvom kontrastného vizuálneho zobrazenia. Tieto stavy sú ľahko identifikovateľné nielen pri priamom pohľade, ale aj v periférnom videní, čo umožňuje užívateľovi sústrediť sa na hranie na klávesy a súčasne monitorovať stav úst.

Zásada spätnej väzby

Po vykonaní akcie (otvorenie alebo zatvorenie úst), užívateľ obdrží okamžitú spätnú väzbu vo forme vizuálnej indikácie na obrazovke, ktorá potvrdzuje zmenu stavu.

Zásada obmedzení

Používateľ je neustále informovaný o tom, či je jeho tvár snímaná a či sú svetelné podmienky dostačujúce pre správne rozpoznávanie

Zásada náznakov

V prípade, že tvár používateľa nie je v zornom poli kamery, aplikácia zobrazí náznak, ktorý povzbudzuje používateľa, aby zariadenie nasmeroval na svoju tvár. Ak je používateľ na správnom mieste, aplikácia zobrazí ústa, ktoré ho nabádajú, aby tie svoje použil na ovládanie aplikácie.

Zásada mapovania

Pohyby úst majú priamy a jednoznačný účinok na používanie sustain pedálu na klavíri. Užívateľ má prirodzený pocit, že ovládanie pedálu je spojené s otváraním a zatváraním úst, čo zjednodušuje porozumenie a používanie systému.

2.1.2 Definovanie interakcie

Po aplikovaní Normanových zásad použiteľnosti, sme definovali konkrétnu interakciu, čo nám pomohlo pri technickom návrhu a implementácii:

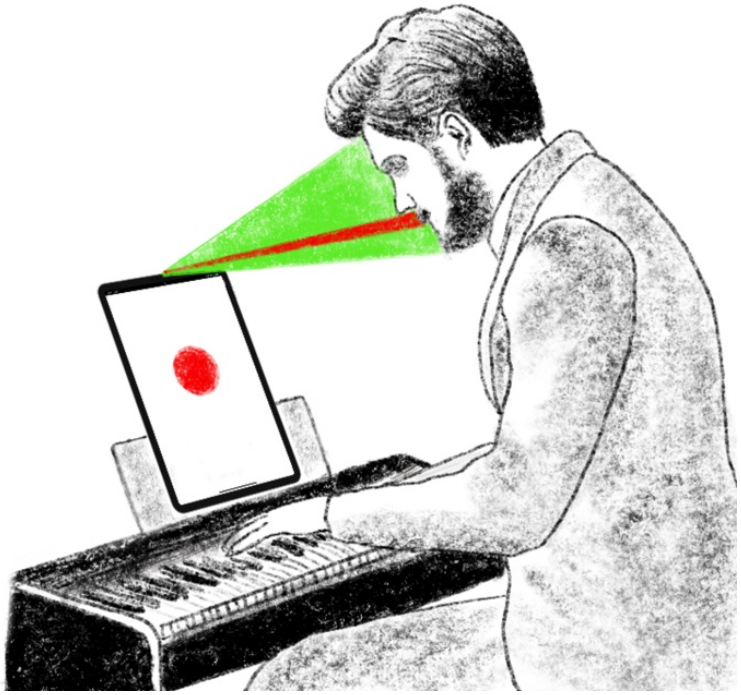
Funkcionalita: Doznenie klavírnych tónov

Rozhranie: Kamery rozpoznávajúce otvorenosť úst

Funkčná časť: Klavírny efekt doznenia

Interakcia: Otvorenie úst – aktivácia pedálu

Úloha používateľa: Ovládanie efektu



Obrázok 55 – Návrh technického prevedenia aplikácie

2.2 Návrh technického riešenia

V tejto kapitole opíšeme výber a overenie technológie využitej pre implementáciu navrhnutej interakcie a komunikácie aplikácie s klavírom.

2.2.1 Návrh technického riešenia interakcie

Najvhodnejšie riešenie sa na základe experimentu ukázalo ovládanie ústami, ktoré musíme technologicky overiť, keďže interakcia vyžaduje jemné rozpoznanie otvorenia úst. Ako experiment sme navrhli viacero jednoduchých prototypov pre testovanie technológie špecificky pre uvedený problém. Natívne komponenty dostupné pre vývojárov ponúkajú dve technológie. Rozpoznávanie z obrazu za pomoci Vision [19] a rozpoznávanie tváre pomocou kombinácie TrueDepth kamery a prednej kamery, ktoré je súčasťou rozšírenej reality frameworku ARKit [20].

Vision

Pre overenie technológie sme implementovali jednoduchý prototyp, ktorý odhalil, že Vision je nespoľahlivé riešenie pre uvedený problém, ktoré by kvôli svojej tendencii nepravidelne prechádzať cez nastavený prah pre spustenie pedálu mohlo prerušiť želané doznenie nôt. Tento problém môže byť vyriešený pomocou spätnej dátovej analýzy, ktorá však zvýšila latenciu, čo je pri hudobnom nástroji nevhodné. Taktiež je tu limitovaná obnovovacia frekvencia na 30 snímok za sekundu.

ARKit

ARKit riešenie dovoľuje nastaviť snímanie na 60 snímok za sekundu, pričom kamere sekunduje TrueDepth kamera. Tá zrejme svojimi približne 15 snímkami za sekundu stabilizuje merania a preto sme zvolili pokračovanie touto metódou. Existuje viacero prístupov, ktoré bolo nutné pre naše riešenie testovať. Porovnali sme odporúčaný vysoko úrovňový prístup pomocou tzv. blendShapes, čo je knižnica pomenovaných koeficientov v ponímaní pohybu špecifických výrazov tváre [21]. Druhá možnosť je pristúpenie k jednotlivému vrcholu na maske tváre. Jeden z vývojárov preskúmal ručne vrcholy masky, aby našiel tie, ktoré patria ústam a medzi ktorými sa nachádza stredný bod pre vrchnú a spodnú peru [22]. Tento prístup nám umožnil sledovať vzdialenosť medzi dvoma bodmi. Prístup rozpoznával jemné pootvorenie úst lepšie než prístup s blendShapes a preto sme ho vybrali ako vhodný pre implementáciu interakcie navrhovaného riešenia.

2.2.2 Návrh technického riešenia komunikácie

Aby bolo naše technické riešenie použiteľné, potrebujeme zabezpečiť komunikáciu medzi aplikáciou a klavírom a celkovo čo najlepšou integráciou riešenia do systému. Možnosti komunikácie sú znázornené na obrázku 16.

MIDI Network

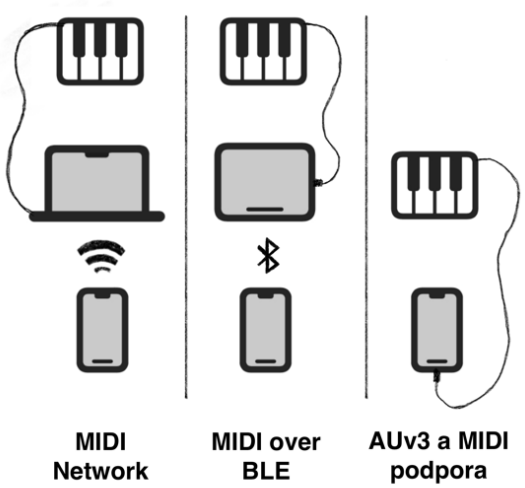
Počítače s operačným systémom macOS môžu využívať natívnu možnosť pre komunikáciu medzi MIDI zariadeniami v rámci lokálnej siete. Tento prístup je podporovaný viacerými aplikáciami zameranými na MIDI. Tieto aplikácie sa môžu pripojiť k Mac počítaču pomocou WiFi, alebo kábla a potom sú MIDI správy dostupné pre ľubovoľný DAW [23]. Implementácia takejto komunikácie vyžaduje využitie knižnicu CoreMIDI, ktorá umožňuje vytváranie a odosielanie MIDI správ v rámci lokálnej siete [24].

MIDI over BLE

Prístup umožňuje odosielanie a prijímanie MIDI správ prostredníctvom bezdrôtovej technológie Bluetooth Low Energy (BLE). Tento prístup umožní využitie aplikácie aj používateľom, ktorí nemajú prístup k počítaču Mac, keďže okrem týchto počítačov je komunikácia MIDI over BLE natívne dostupná aj na iOS a iPadOS. Vďaka tomu by sme umožnili prepojenie aplikácie napríklad so zariadením iPad, kde bezplatná aplikácia z produkcie Apple s názvom GarageBand umožňuje pripojenie príslušenstva s podporou MIDI over BLE [25]. Na ostatných operačných systémoch ako je Windows, Linux či Android je nutné nainštalovať potrebné ovládače, či aplikácie. Natívna knižnica CoreBluetooth dovoľuje aplikácii tváriť sa ako BLE príslušenstvo [26].

AUv3

Pre produkciu hudby sa často používajú nielen zariadenia s operačným systémom macOS, ale aj iPadOS, či iOS. Komunita používateľov a vývojárov formovala počas existencie iOS zaujímavú architektúru v dnešnej forme s názvom AUv3, ktorá umožňuje vytvorenie aplikácie, ktorú je možné využiť samostatne, alebo je možné jej nástroje a rozhranie načítať v inej aplikácii [27]. Na túto problematiku sa môžeme pozrieť aj z druhej strany a to implementáciou samotného hosťovania cudzích aplikácií v tej našej. Čiže rovnaká funkcionalita ako pri aplikácii GarageBand, ktorá dokáže AUv3 nástroje načítať a ovládať. Architektúra AUv3 umožňuje pre nami riešený problém vytvoriť samostatné riešenie, ktoré nevyžaduje počítač, či tablet. Pre ovládanie načítaného nástroja hosťovanej aplikácie sa ako vhodné riešenie ukazuje využitie knižnice AudioKit, vďaka ktorej dokážeme prijímať MIDI správy z pripojených MIDI kláves [28].



Obrázok 66 – Metódy komunikácie medzi nástrojom a aplikáciou

Všetky tieto prístupy majú určité výhody a nevýhody. Pri výbere záleží na používateľovi a jeho preferenciách. Preto navrhujeme a implementujeme všetky opisované možnosti komunikácie.

Latencia komunikácie

Považovali sme za vhodné porovnať latenciu riešenia MIDI siete a MIDI over BLE. Vďaka aplikácii MIDI Monitor [29] sme spoľahlivo dokázali odmerať rozdiel v latencii, kde sme sledovali čas prijatia MIDI správy odoslanej z aplikácie. Z našej aplikácie sme naraz poslali správu prostredníctvom oboch spôsobov komunikácie a v MIDI Monitor aplikácii sme sledovali čas prijatia (obrázok 17). Toto porovnanie nám ukázalo, že komunikácia cez MIDI sieť je konštantne rýchlejšia než BLE v priemere o 11 milisekúnd. Preto budeme v aplikácii nabádať používateľa k využitiu MIDI siete pre získanie nižšej latencie, ak má k dispozícii Mac počítač.

14:31:38.527	From Sieť Relácia	Control	1	Damper Pedal (Sustain)	127
14:31:38.539	From MouthPedal	Control	1	Damper Pedal (Sustain)	127
14:31:38.561	From Sieť Relácia	Control	1	Damper Pedal (Sustain)	0
14:31:38.569	From MouthPedal	Control	1	Damper Pedal (Sustain)	0
14:31:38.593	From Sieť Relácia	Control	1	Damper Pedal (Sustain)	127
14:31:38.599	From MouthPedal	Control	1	Damper Pedal (Sustain)	127

Obrázok 77 – Zaznamenávanie času prijatia MIDI správy v aplikácii MIDI Monitor

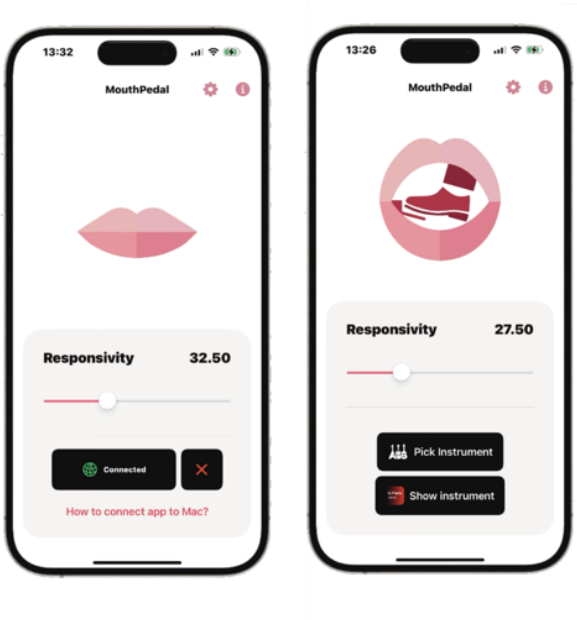
2.3 Návrh grafického používateľského rozhrania

Pri návrhu grafického používateľského rozhrania sme sa riadili návrhom interakcie s podporou Normanových zásad použiteľnosti a odporúčaní pre tvorenie používateľského rozhrania spoločnosti Apple [30]. Framework SwiftUI [31] dovoľuje rýchlo navrhovať

a v krátkych interáciách meniť výzor a funkčnosť používateľského rozhrania, vďaka čomu nebolo potrebné vytvárať grafické návrhy. Ihneď bolo jasné, ako presne bude rozhranie pracovať a vyzerať. Apple tiež ponúka pre vývojárov grafické symboly, ktoré môžu jednoducho využiť pri tvorbe používateľského rozhrania vďaka priamej podpore SwiftUI. Katalóg symbolov sa nachádza v aplikácii pre Mac s názvom SFSymbols [32].

2.3.1 Hlavná obrazovka

Na obrázku 18 sa nachádza hlavná obrazovka, ktorej dominantný prvok používateľovi napovedá, či sú jeho ústa pootvorené, alebo zavreté. Podľa otvorenosti úst sa zobrazujú buď zavreté ústa, alebo otvorené ústa, v ktorých je vyobrazený stlačený pedál. Používateľ týmto získava spätnú väzbu, čo posilňuje interakciu a používateľovu adaptáciu. V dolnej časti obrazovky sa nachádza posuvník, ktorým používateľ ovláda prah otvorenosti úst pre aktiváciu pedálu. Pod posuvníkom sme umiestnili prvok informujúci o stave pripojenia v kontexte využitia MIDI over BLE a MIDI Network metód. V kontexte AUv3 metódy je v tejto časti zobrazené tlačidlo pre zobrazenie výberu nástroja a tlačidlo pre zobrazenie grafického rozhrania samotného nástroja. V pravej hornej časti obrazovky sa už nachádza len tlačidlo pre zobrazenie popisu o aplikácii a tlačidlo pre zobrazenie nastavení.

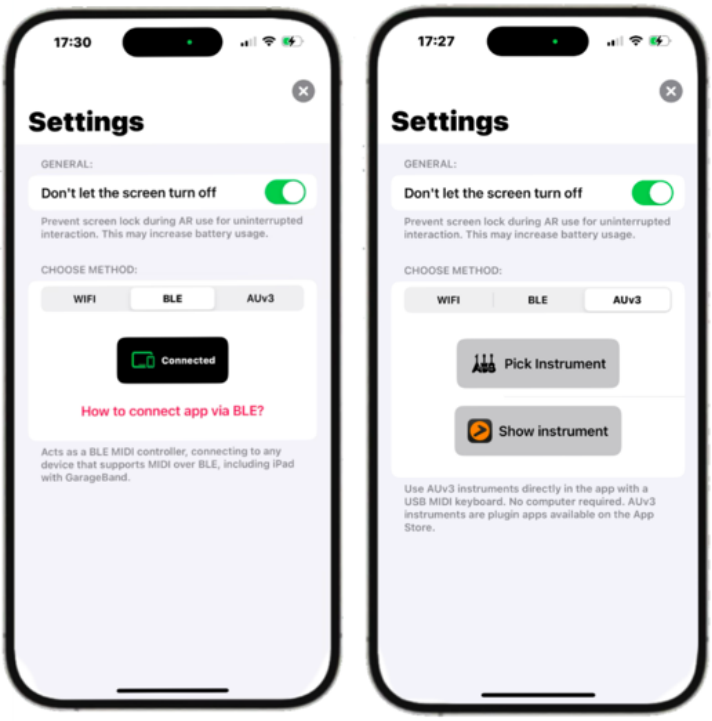


Obrázok 88 – Hlavná obrazovka aplikácie

2.3.2 Nastavenia aplikácie

Pri navrhnutej interakcii sa používateľ počas samotného používania nedotýka displeja, čo by po čase mohlo spôsobiť zamknutie obrazovky. Preto ponúkame možnosť vypnúť časovač

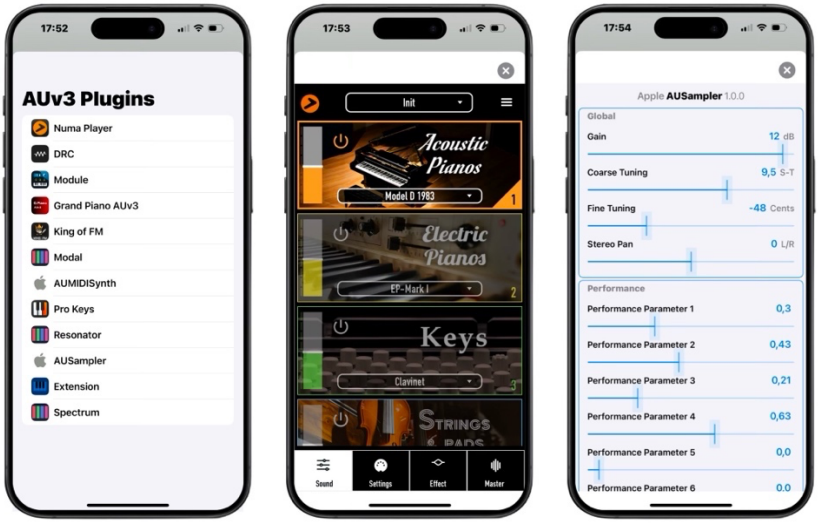
uzamknutia displeju [33]. V ďalšej časti obrazovky sa nachádza možnosť výberu metódy využitia. Pri každej metóde je krátky popis a návod na obsluhu. Nastavenia aplikácie je možné vidieť na obrázku 19.



Obrázok 99 – Obrazovka nastavení aplikácie

2.3.3 Obrazovky pre manažment AUv3 pluginu

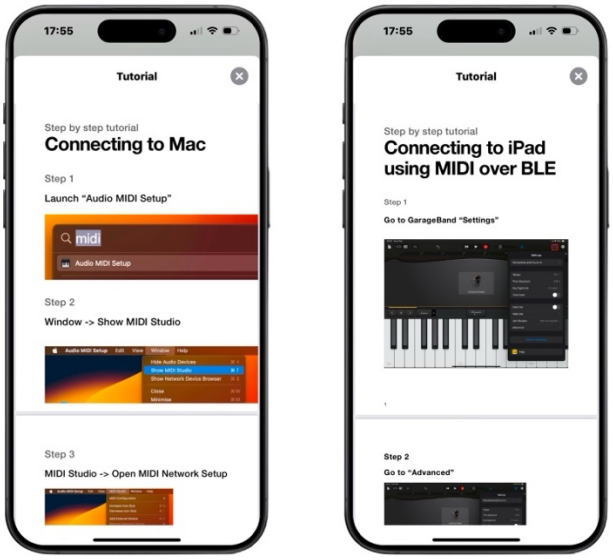
Pre používateľovu voľbu AUv3 pluginu sme navrhli obrazovku, ktorá vykresľuje každý dostupný AUv3 plugin v podobe jeho ikony a názvu. Načítaný nástroj je často nutné konfigurovať pre želaný výstup hudobníka a preto umožňujeme zobrazenie obrazovky aplikácie tretej strán, ktorá plugin ponúka. Počas behu aplikácie si používateľ môže zmeniť nástroj, či predvoľbu pluginu alebo vybrať iný plugin. V popise funkcionality informujeme používateľa o dostupnosti pluginov v obchode Apple AppStore. Obrazovky pre manažment AUv3 pluginov sa nachádzajú na obrázku 20



Obrázok 20 – Obrazovky pre prácu s AUv3 pluginmi

2.3.4 Návod na použitie

Postup pre pripojenie MIDI Network kompatibilnej aplikácie ku počítaču Mac, či pripojenie pomocou BLE nie je veľmi priamočiary. Preto v našej aplikácii ponúkame obrazovku (obrázok 21), ktorá používateľovi zobrazí podrobný návod vo formáte PDF, ktorý sa nachádza v prílohách A a B.



Obrázok 21 – Obrazovka návodu na pripojenie

2.4 Implementácia riešenia

V tejto kapitole je opísaná implementácia interakcie a komunikácie navrhnutého riešenia.

2.4.1 Implementácia interakcie

V časti kódu na obrázku 22, ktorú sme implementovali, využívame knižnicu ARKit na sledovanie a interpretáciu pohybu tváre. Po aktualizácii ARSession a príchode nových záznamov kotiev sa funkcia session(_:didUpdate:) zameriava na spracovanie týchto údajov. Počiatočným krokom je identifikácia prvej kotvy reprezentujúcej tvár pomocou ARFaceAnchor. Pokiaľ je táto identifikácia úspešná, pokračujeme získavaním súradníc vrcholov tváre, ktoré nám umožňujú lokalizovať polohu hornej a dolnej pery. Využívame funkciu simd_distance() na výpočet vzdialenosti medzi hornou a dolnou perou, čo nám poskytuje informáciu o stupni otvorenia úst [34]. Ak táto vzdialenosť prekročí stanovený prah, vykonávame príslušné akcie na aktiváciu pedálu. V opačnom prípade, ak je táto vzdialenosť pod stanoveným prahom, pedál zostáva neaktivovaný. V každom kroku tohto procesu sledujeme stav úst a na základe toho sa rozhodujeme, či pedál aktivujeme alebo deaktivujeme.

```
func session(_ session: ARSession, didUpdate anchors: [ARAnchor]) {
    guard let faceAnchor = anchors.first(where: { $0 is ARFaceAnchor }) as?
        ARFaceAnchor else { return }

    let vertices = faceAnchor.geometry.vertices

    let upperLipPosition = vertices[24]
    let lowerLipPosition = vertices[25]

    let lipsOpenness = simd_distance(upperLipPosition, lowerLipPosition)

    if lipsOpennnes > Float(self.arViewContainer.trashold) {
        // deaktivácia pedálu
        print("SustainOff")
    }else {
        // aktivácia pedálu
        print("SustainOn")
    }
}
```

Obrázok 22 – Implementácia ARKit interakcie

Aby sme mohli používateľa informovať o tom, či je jeho tvár snímaná a svetelnosť priaznivá, využívame vstavané metódy ARKit frameworku (obrázok 23)

```
self.arViewContainer.lightEstimate = frame.lightEstimate?.ambientIntensity ?? 0
self.arViewContainer.isFaceTracked = faceAnchor.isTracked
```

Obrázok 23 – Stav osvetlenia tváre a jej snímania

2.4.2 Implementácia komunikácie

MIDI Network

Metóda setupMIDINetworkSession() (obrázok 24) v triede MIDIManager hrá kľúčovú úlohu pri inicializácii a konfigurácii sieťovej MIDI relácie. Najprv vytvára nového klienta MIDI pomocou funkcie MIDIClientCreateWithBlock(), ktorý je potom pripojený k systému MIDI [35]. Tento klient je identifikovaný reťazcom "MIDIManagerClient". Ďalej nasleduje vytvorenie výstupného portu MIDI s funkciou MIDIOutputPortCreate(), ktorý umožňuje triede MIDIManager odosielať MIDI správy [36]. Potom metóda získava inštanciu predvolenej sieťovej MIDI relácie pomocou MIDINetworkSession.default(), čo je nevyhnutné pre komunikáciu s ostatnými MIDI zariadeniami v sieti [37]. Relácia je následne aktivovaná a nastavená politika pripojenia na "anyone", čo umožňuje prijímať spojenia od ľubovoľného zariadenia v sieti. Nakoniec metóda kontroluje a odstraňuje všetky existujúce pripojenia v sieti, čím sa zabezpečí, že po inicializácii siete nebudú žiadne staré spojenia, ktoré by mohli spôsobiť konflikty.

```
private func setupMIDINetworkSession() {
    MIDIClientCreateWithBlock("MIDIManagerClient" as CFString, &client, nil)
    MIDIOutputPortCreate(client, "MIDIManagerOutputPort" as CFString, &outputPort)

    midiNetworkSession = MIDINetworkSession.default()

    midiNetworkSession?.isEnabled = true
    midiNetworkSession?.connectionPolicy = .anyone

    if let connections = midiNetworkSession?.connections() {
        for connection in connections {
            midiNetworkSession?.removeConnection(connection)
        }
    }
}
```

Obrázok 24 – Nastavenie MIDI Network relácie

Metóda sendControlChange(controller: value:) v triede MIDIManager je zodpovedná za vytvorenie MIDI správy. V tejto metóde sa najprv vytvára štruktúra Data, do ktorej sa postupne pridávajú jednotlivé bajty reprezentujúce kontrolnú správu. Po vytvorení MIDI dát sa volá metóda sendMIDIData(data:), ktorá zabezpečuje ich odoslanie cez MIDI rozhranie.

```
func sendControlChange(controller: UInt8, value: UInt8) {
    let channel: UInt8 = 0
    let controlChangeStatus: UInt8 = 0xB0 | (channel & 0x0F)

    var data = Data()
    data.append(controlChangeStatus)
    data.append(controller & 0x7F)
    data.append(value & 0x7F)

    sendMIDIData(data: data)
}
```

Obrázok 25 – Vytvorenie MIDI správy pre odoslanie

Metóda sendMIDIData(data:) v triede MIDIManager slúži na odoslanie MIDI dát. Najprv sa určí cieľová destinácia pomocou funkcie MIDIGetDestination(0). Ak je destinácia platná (tj. nerovná sa 0), vytvorí a inicializuje sa packet MIDI dát. Následne sa zistí dĺžka dát pomocou data.count a pridajú sa do balíka pomocou MIDIPacketListAdd(packetList, 1024, packet, 0, dataLength, unsafeBytes) [38]. Nakoniec sa vytvorené MIDI dáta odosielajú pomocou MIDISend(outputPort, destination, packetList) [39]. Na záver sa uvoľní pamäť balíka pomocou packetList.deallocate(). Implementácia sa nachádza na obrázku 26.

```
private func sendMIDIData(data: Data) {
    let destination = MIDIGetDestination(0)
    if destination != 0 {
        let packetList = UnsafeMutablePointer<MIDIPacketList>.allocate(capacity: 1)
        let packet = MIDIPacketListInit(packetList)
        let dataLength = data.count
        data.withUnsafeBytes { (bytes: UnsafeRawBufferPointer) in
            let unsafeBytes = bytes.bindMemory(to: UInt8.self).baseAddress!
            MIDIPacketListAdd(packetList, 1024, packet, 0, dataLength, unsafeBytes)
        }
        MIDISend(outputPort, destination, packetList)
        packetList.deallocate()
    }
}
```

Obrázok 26 – Odoslanie MIDI správy

Metóda startConnectionStatusTimer() v triede MIDIManager slúži na spustenie časovača, ktorý pravidelne aktualizuje stav pripojenia. V tejto metóde sa vytvára opakujúci sa časovač s intervalom jednej sekundy, ktorý volá metódu updateConnectionStatus(). Táto metóda je zabezpečená ako slabá referencia pre zabránenie úniku pamäte. Metóda updateConnectionStatus() slúži na aktualizáciu stavu pripojenia. V tejto metóde sa zisťuje, či je zoznam pripojený MIDI sieťovej relácie prázdny. Ak áno, premenná isConnected sa nastaví na false, inak sa nastaví na true. Tento proces je vykonávaný na hlavnom vlákne, aby sa zabezpečila správna synchronizácia s GUI. Implementácia sa nachádza na obrázku 27.

```
private func startConnectionStatusTimer() {
    DispatchQueue.main.async {
        self.connectionStatusTimer = Timer.scheduledTimer(withTimeInterval: 1.0, repeats: true) { [weak self] _ in
            self?.updateConnectionStatus()
        }
    }

}

private func updateConnectionStatus() {
    DispatchQueue.main.async {
        self.isConnected = !(self.midiNetworkSession?.connections().isEmpty ?? true)
    }
}
```

Obrázok 27 – Získavanie stavu MIDI Network pripojenia

MIDI over BLE

Pre vývojárov je pripravená hotová obrazovka v podobe ViewControlleru, ktorá slúži pre manažment spustenie a sledovanie stavu pripojenia komunikácie MIDI over BLE [40]. Po našom testovaní sme identifikovali nesprávnu funkčnosť, rovnako ako aj u ostatných dostupných aplikáciách, ktoré túto obrazovku implementujú. Chybovosť spočíva v nepredvídateľnom správaní grafického rozhrania a základnej funkčnosti. Implementáciu natívnej obrazovky je možné vidieť na obrázku 28.

```
struct CABTMIDILocalPeripheralView: UIViewControllerRepresentable {

    func makeUIViewController(context: Context) -> UIViewController {
        let midiViewController = CABTMIDILocalPeripheralViewController()

        return midiViewController
    }

    func updateUIViewController(_ uiViewController: UIViewController, context: Context) {}
}
```

Obrázok 28 – Natívna implementácia MIDI over BLE obrazovky

Kvôli chybnému správaniu sa natívnej obrazovky sme sa rozhodli implementovať vlastnú obrazovku, čo však vyžaduje vlastnú implementáciu manažovania Bluetooth komunikácie, keďže tá bola v obrazovke už implementovaná a tým pádom schovaná pred vývojárom. V kóde na obrázku 29 nastavujeme službu Bluetooth MIDI pomocou frameworku CoreBluetooth. Začína tým, že definuje UUID (univerzálne jedinečné identifikátory) pre MIDI službu a charakteristiku. Tieto UUID jednoznačne identifikujú rôzne typy služieb a charakteristík Bluetooth Low Energy (BLE). Ďalej je vytvorená inštancia CBMutableCharacteristic pre MIDI charakteristiku [41]. Potom je vytvorená inštancia CBMutableService pre MIDI službu pomocou preddefinovaného UUID. Ďalej sú charakteristiky služby nastavené na vytvorenú MIDI charakteristiku. Potom sa skontroluje

stav periférneho manažéra, inštancie CBPeripheralManager [42]. Ak je zapnutý, MIDI služba je pridaná k perifémemu manažérovi pomocou metódy add(_). Nasledujúcim krokom je vytvorenie údajov pre publikovanie Bluetooth pomocou slovníka advertisementData. Obsahuje názov lokálneho zariadenia a identifikátor služby. Názov zariadenia je získaný pomocou UIDevice.current.name, zatiaľ čo identifikátor služby je UUID MIDI služby. Tieto údaje sú potom nastavené ako publikačné údaje pri spustení publikovania metódou startAdvertising(advertisementData), čo umožňuje iným zariadeniam nájsť a pripojiť sa k tomuto zariadeniu pomocou Bluetooth [43]. Posielanie MIDI správ je možné implementovať pomocou kombinácie využitia knižníc CoreBluetooth a CoreMidi. Knižnica AudioKit však ponúka jednoduchší, vysoko úrovňový prístup. Metóda openOutput(name: "Bluetooth") otvorí Bluetooth výstup pre odosielanie MIDI správ. Implementácia sa nachádza na obrázku 29.

```
func setupMIDIService() {
    let midiServiceUUID = CBUUID(string: "03B80E5A-EDE8-4B33-A751-6CE34EC4C700")
    let midiCharacteristicUUID = CBUUID(string: "7772E5DB-3868-4112-A1A9-F2669D106BF3")

    midiCharacteristic = CBMutableCharacteristic(type: midiCharacteristicUUID,
                                                 properties: [.notify, .writeWithoutResponse],
                                                 value: nil, permissions: [.readable, .writeable])

    midiService = CBMutableService(type: midiServiceUUID, primary: true)
    midiService?.characteristics = [midiCharacteristic!]

    if peripheralManager.state == .poweredOn {
        peripheralManager.add(midiService!)
    }

    let advertisementData = [
        CBAdvertisementDataLocalNameKey: UIDevice.current.name,
        CBAdvertisementDataServiceUUIDsKey: [self.midiService!.uuid]
    ] as [String : Any]

    peripheralManager.startAdvertising(advertisementData)

    midi.openOutput(name: "Bluetooth")

}
```

Obrázok 29 – Nastavenie BLE služby

Metódy na obrázku 30 slúžia na ovládanie sustain pedálu. Prvá metóda sustainOn nastavuje pedál sustain do stavu "zapnutý". Vytvára správu ovládača MIDI, ktorá posiela príkaz na zapnutie pedálu sustain na zadanom MIDI kanáli. Pedál sustain je ovládaný pomocou ovládača s číslom 64, a hodnota 127 znamená, že pedál je stlačený. Druhá metóda sustainOff nastavuje pedál sustain do stavu "vypnutý". Podobne ako prvá funkcia, vytvára správu ovládača MIDI, ktorá posiela príkaz na vypnutie pedálu sustain na zadanom MIDI kanáli. Opäť je použitý ovládač s číslom 64 a hodnota 0 označuje, že pedál je uvoľnený.

```swift
func sustainOn(channel: MIDIChannel = 0) {
    let sustainControllerNumber = MIDIByte(64)
    let sustainOnValue = MIDIByte(127)

    midi.sendControllerMessage(sustainControllerNumber, value: sustainOnValue, channel: channel)
}

func sustainOff(channel: MIDIChannel = 0) {
    let sustainControllerNumber = MIDIByte(64)
    let sustainOffValue = MIDIByte(0)

    midi.sendControllerMessage(sustainControllerNumber, value: sustainOffValue, channel: channel)
}
```

Obrázok 30 – Vytvorenie a odoslanie MIDI správy

Metóda na obrázku 31 startConnectionStatusTimer() inicializuje a spúšťa časovač na sledovanie stavu pripojenia. V rámci tejto metódy sa vytvára opakujúci sa časovač s intervalom 1 sekundy, ktorý vykonáva metódu updateConnectionStatus(). Tento časovač je priraďovaný ako slabá referencia, aby sa zabránilo možnému úniku pamäte. Metóda updateConnectionStatus() slúži na aktualizáciu stavu pripojenia. V rámci tejto metódy sa najprv zisťuje, či existuje pripojenie s názvom "Bluetooth" v zozname dostupných destinácií MIDI. Ak áno, tak sa premenná isConnected nastaví na hodnotu true. Okrem toho sa zastavuje publikovanie, aby sa zabránilo pripojeniu iného zariadenia. Ak nie je zistené pripojenie cez Bluetooth, premenná isConnected sa nastaví na false. Tento proces je vykonávaný na hlavnom vlákne, aby sa zabezpečila správna synchronizácia s GUI, kde používateľa o stave pripojenia informujeme.

```swift
private func startConnectionStatusTimer() {
    DispatchQueue.main.async {
        self.connectionStatusTimer = Timer.scheduledTimer(withTimeInterval: 1.0, repeats: true) { [weak self] _ in
            self?.updateConnectionStatus()
        }
    }

}

private func updateConnectionStatus() {
    DispatchQueue.main.async {
        if self.midi.destinationNames.contains("Bluetooth"){
            if self.isConnected{
                if self.peripheralManager.state == .poweredOn {
                    self.peripheralManager.stopAdvertising()
                    print("stopped")
                }
            }
            self.isConnected = true

        }else{
            self.isConnected = false
        }
    }
}
```

Obrázok 31 – Získavanie stavu BLE pripojenia

AUv3

Metóda na obrázku 32 umožňuje načítanie AUv3 nástroja, ktorý je následne ovládaný pomocou MIDI klávesov pripojených priamo k zariadeniu, kde je spustená aplikácia MouthPedal. Prvý krok pri práci s AUv3 komponentami je ich načítanie, počas ktorého sa vyhľadajú všetky AUv3 nástroje z aplikácii tretích strán na zariadení. Kód na obrázku 6 ukazuje, že najprv vytvárame singleton inštanciu triedy AVAudioUnit ComponentManager, ktorý slúži ako správca AUv3 nástrojov a efektov [44]. Z dostupných komponentov vyberáme typ kAudioUnitTypeMusicDevice, vďaka čomu získame len nástroje [45]. Následne si nástroje uchováme do poľa typu AUv3Pluging, ktorý sme si definovali, aby sme uchovávali potrebné dáta pre vykreslenie v SwiftUI a taktiež id nástroja, aby sme sa k nemu mohli dostať neskôr pri zapojení nástroja do grafu našej aplikácie. Nástroje obsahujú aj ikonu aplikácie, z ktorej nástroj pochádza, ktorú uchovávame pre zobrazenie používateľovi pre lepšie rozpoznanie.

```
func loadAvailableInstruments() {
    let componentManager = AVAudioUnitComponentManager.shared()
    let components = componentManager.components(matching: .init(componentType: kAudioUnitType_MusicDevice,
                                                                 componentSubType: 0,
                                                                 componentManufacturer: 0,
                                                                 componentFlags: 0,
                                                                 componentFlagsMask: 0))
    DispatchQueue.main.async {
        self.plugins.removeAll()
        for component in components {
            let plugin = AUv3Plugin(id: UUID(),
                                    name: component.name,
                                    manufacturer: component.manufacturerName,
                                    isSelected: false,
                                    image: component.icon)
            self.plugins.append(plugin)
        }
    }
}
```

Obrázok 32 – Načítanie dostupných AUv3 plugin nástrojov

Po používateľovom výbere nástroja ho môžeme použiť vytvorením inštancie vybraného nástroja, aby sme ho mohli zapojiť do audio grafu ako uzol audioUnit. Tento uzol následne zapájame do hlavného mixéru [46], ktorý je už predvolene pripojený na výstup. Následne môžeme spustiť audio engine [47] a načítať prvý preset nástroja [48]. Implementácia sa nachádza na obrázku 33.

```
AVAudioUnit.instantiate(with: component.audioComponentDescription, options: []) { [weak self] (audioUnit, error) in

    DispatchQueue.main.async {
        guard let self = self, let audioUnit = audioUnit, error == nil else {
            print("Error instantiating the audio unit: \(error?.localizedDescription ?? "Unknown error")")
            return
        }

        self.selectedInstrument = audioUnit
        self.audioEngine.attach(audioUnit)
        self.audioEngine.connect(audioUnit, to: self.audioEngine.mainMixerNode, format: nil)

        if !self.audioEngine.isRunning {
            do {
                try self.audioEngine.start()
            } catch {
                print("Could not start audio engine: \(error)")
            }
        }

        let auAudioUnit = audioUnit.auAudioUnit
        if let factoryPresets = auAudioUnit.factoryPresets, !factoryPresets.isEmpty {
            auAudioUnit.currentPreset = factoryPresets.first
            print("Factory preset loaded successfully.")
        } else {
            print("No factory presets available or unable to access them.")
        }
```

Obrázok 33 – Zapojenie vybraného AUv3 pluginu do audio grafu

Aby sme mohli načítaný a zapojený nástroj ovládať, musíme byť schopní spustiť, či zastaviť prehrávanie nôt a taktiež aktivovať, alebo deaktivovať pedál. Chceme ovládať nástroj prostredníctvom pripojeného MIDI klavíru a preto na obrázku 34 ukazujeme implementáciu delegát metódy protokolu MIDIListener knižnice AudioKit [49]. Tieto metódy sú volané pri každej správe prijatej z MIDI zariadenia. Implementovaná je metóda *recievedMIDINoteOn,* z ktorej si vieme prečítať číslo spustenej noty a silu stlačenia, čo je nutné pre adekvátne spustenie noty načítaného nástroja. Metóda *recievedMIDINoteOff* nám povie, ktorá nota bola na MIDI zariadení vypnutá, vďaka čomu vieme vypnúť danú notu nástroja.

```
func receivedMIDINoteOn(noteNumber: AudioKit.MIDINoteNumber,
                        velocity: AudioKit.MIDIVelocity,
                        channel: AudioKit.MIDIChannel,
                        portID: MIDIUniqueID?,
                        timeStamp: MIDITimeStamp?) {
    //spustenie noty
    print("noteOn")
}

func receivedMIDINoteOff(noteNumber: AudioKit.MIDINoteNumber,
                         velocity: AudioKit.MIDIVelocity,
                         channel: AudioKit.MIDIChannel,
                         portID: MIDIUniqueID?,
                         timeStamp: MIDITimeStamp?) {
    //vypnutie noty
    print("noteOff")
}
```

Obrázok 34 – Delegát metódy pre detekciu interakcie s MIDI klávesmi

Implementované delegát metódy volajú spustenie a zastavenie noty. Aktiváciu a deaktiváciu pedálu voláme na základe logiky v ARKit interakcii. Hodnota aktivovaného pedálu je 127 a hodnota deaktivovaného pedálu je 0. Tieto volania ovládania nástroja môžeme vidieť na obrázku 35.

```
// spustenie noty nástroja
instrument.startNote(noteNumber, withVelocity: velocity, onChannel: channel)
// zastavenie noty nástroja
instrument.stopNote(noteNumber, onChannel: channel)
// aktivácia pedálu
instrument.sendController(64, withValue: 127, onChannel: channel)
// deaktivácia pedálu
instrument.sendController(64, withValue: 0, onChannel: channel)
```

Obrázok 35 – Ovládanie načítaného AUv3 plugin nástroja

2.4.3 Architektúra aplikácie

V časti SwiftUI kódu sú definované pozorovateľné objekty pre správcu AUv3 (hostManager), správcu MIDI Network (midiNetoworkManager) a správcu Bluetooth MIDI (bleManager). Okrem toho sú tu aj definované stavové premenné, ako napríklad úroveň otvorenia úst (mouthOpenness) a prahová hodnota (trashold). Grafické zobrazenie ARViewContainera sme využívali len vo fáze vývoja a preto používame modifikátor .oppacity(), ktorým prvok zneviditeľníme.

```
struct ContentView: View {
    @ObservedObject var hostManager = AUv3Host()
    @ObservedObject var midiNetworkManager = MIDIManager()
    @ObservedObject var bleManager = BluetoothMIDImanager()

    @State private var mouthOpenness: Float = 0.0
    @AppStorage("trashold") private var trashold: Double = 0.003

    var body: some View {
        if trueDepthSupported{
            ZStack{
                VStack {
                    ARViewContainer(mouthOpenness: $mouthOpenness,
                                    midiNetworkManager: midiNetworkManager,
                                    hostManager: hostManager,
                                    bleManager: bleManager).opacity(0.0)
```

Obrázok 36 – SwiftUI časť architektúry

V rámci architektúry aplikácie je implementovaná technika dependency injection na umožnenie komunikácie medzi SwiftUI, ARKit a implementovanými modulmi. Štruktúra ARViewContainer obaľuje AR pohľad a poskytuje jeho riadenie. Obsahuje referencie na správcov MIDI, AUv3 a Bluetooth MIDI, ktoré sú predané ako závislosti cez konštruktor. Coordinator riadi AR reláciu a zabezpečuje komunikáciu medzi AR zobrazením a zvyškom aplikácie. Coordinator je trieda, ktorá implementuje ARSessionDelegate a má prístup k rodičovskej ARViewContainer štruktúre, ako aj k manažérom MIDI, AUv3 a Bluetooth MIDI. Tento prístup umožňuje jednoduchú integráciu a ovládanie troch modulov z oboch prostredí - SwiftUI a ARKit a poskytuje tak vyváženú a rozšíriteľnú architektúru pre aplikáciu.

```swift
struct ARViewContainer: UIViewRepresentable {
    @Binding var mouthOpenness: Float
    @Binding var trashold: Double

    var midiNeetworkManager: MIDIManager
    var hostManager: AUv3Host
    var bleManager: BluetoothMIDImanager

    func makeCoordinator() -> Coordinator {
        Coordinator(self, midiManager: midiManager, hostManager: hostManager, bleManager: bleManager)
    }

    class Coordinator: NSObject, ARSessionDelegate {
        var arViewContainer: ARViewContainer

        var midiNetworkManager: MIDIManager
        var hostManager: AUv3Host
        var bleManager: BluetoothMIDImanager

        init(_ arViewContainer: ARViewContainer, midiManager: MIDIManager, hostManager: AUv3Host, bleManager: BluetoothMIDImanager) {
            self.arViewContainer = arViewContainer

            self.midiNetworkManager = midiManager
            self.hostManager = hostManager
            self.bleManager = bleManager

            super.init()
        }
```

Obrázok 37 – Dependancy injection

3 Nasadenie riešenia

V životnom cykle každej aplikácie sa odráža dynamický proces, ktorým prechádza od jej počiatočného konceptu a vývoja až po nasadenie a udržiavanie v prevádzke. Avšak, táto cesta sa môže dramaticky líšiť v závislosti od toho, či je aplikácia vyvíjaná jednotlivcom, alebo veľkou firmou. Pre jednotlivca sú dôležité nielen technické zručnosti a tvorivosť, ale aj schopnosť efektívne riadiť všetky aspekty životného cyklu aplikácie, vrátane plánovania, vývoja, testovania a nasadenia. Na druhej strane, veľké firmy disponujú väčšími finančnými prostriedkami, ľudskými zdrojmi a infraštruktúrou, ktoré im umožňujú vykonávať tieto úlohy na rozsiahlejšej úrovni.

3.1 Launch as MVP

Spustenie minimálneho životaschopného produktu (MVP) je stratégia, pri ktorej sa aplikácia spúšťa na trh s minimálnou súpravou funkcií a vlastností, ktoré sú nevyhnutné na splnenie základných potrieb užívateľov. Cieľom je čo najrýchlejšie a najlacnejšie získať spätnú väzbu od skutočných používateľov a použiť ju na ďalšie vylepšenie a rozšírenie aplikácie. Pri spustení MVP sa vývojári zameriavajú na to, aby ich aplikácia obsahovala iba základné funkcionality, ktoré umožnia užívateľom vykonávať primárne úlohy, alebo dosahovať primárne ciele. Spustenie ako MVP umožňuje vývojárom ušetriť čas a zdroje tým, že sa vyhnú nadmernému vývoju funkcií, ktoré by mohli byť zbytočné, alebo nevyžiadané zo strany užívateľov. Vývojári môžu neskôr postupne pridávať ďalšie funkcie a vlastnosti na základe spätnej väzby od užívateľov a analýzy trhu. Týmto spôsobom sa aplikácia postupne rozširuje, zdokonaľuje a prispôsobuje sa potrebám a požiadavkám svojich užívateľov [50].

3.2 Definovanie MVP

Definovanie MVP pre spustenie produktu na trhu sa od klasického definovania MVP líši tým, že MVP pre spustenie na trh musí byť plne funkčné riešenie a taktiež vyžaduje prácu naviac súvisiacu s náležitosťami spojenými s publikovaním. Medzi tieto náležitosti patrí splnenie predpisov, ktoré stanovuje spoločnosť prevádzkujúca obchod s aplikáciami. Následne je nutné vytvoriť popis a snímky obrazovky aplikácie, ktoré by mali pomôcť používateľom pochopiť schopnosti aplikácie aj bez jej stiahnutia. Prehľad funkcionalít, ktoré sme vybrali ako nutné pre funkčnosť a použiteľnosť uvádzame v tabuľke 3.

Popis funkcionality
Rozpoznávanie otvorenosti úst
Nastavenie prahu otvorenosti úst pre spustenie pedálu
Komunikácia s DAW prostredníctvom MIDI Network
Návod pre nastavenie MIDI Network komunikácie
Možnosť vypnutia časovaču nečinnosti
Náležitosti spojené s publikovaním aplikácie

Tabuľka 3 – Definovanie MVP funkcionalít

3.3 Publikovanie aplikácie v AppStore

Publikovanie aplikácie Apple App Store predstavuje dôležitý krok pre vývojárov, ktorí chcú svoje produkty sprístupniť pre milióny užívateľov iOS zariadení. Proces začína vytvorením vývojárskeho účtu v Apple Developer Program, kde je možné registrovať a spravovať aplikácie. Po dokončení vývoja a testovania je potrebné vytvoriť profil aplikácie a dodržať štandardy týkajúce sa dizajnu, funkčnosti a obsahu. Po schválení aplikácie moderátormi App Store môže byť publikovaná a dostupná na stiahnutie pre užívateľov z celého sveta. Tento proces zabezpečuje kvalitu a bezpečnosť aplikácií a poskytuje vývojárom prístup k širokej a globálnej zákazníckej základne [51].

3.3.1 AppStore Connect

App Store Connect je nástroj od spoločnosti Apple, ktorý poskytuje vývojárom možnosť spravovať ich aplikácie pred ich uvedením na trh v App Store, ako aj po ich publikovaní. Umožňuje im sledovať výkonnosť aplikácií pomocou štatistík týkajúcich sa sťahovaní, príjmov a hodnotení od užívateľov [52].

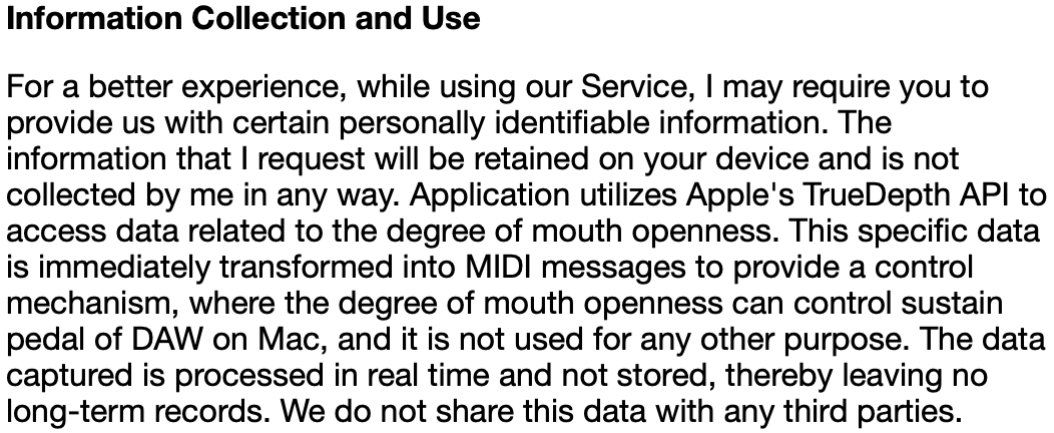
3.3.2 Zamietnutie publikovania

Prvá verzia, ktorú sme poslali pre overenie bola zamietnutá pre publikovanie. Pre pokračovanie v procese si AppStore tím vyžiadal video, ktoré dokumentuje správnu funkčnosť interakcie a vyžiadanie povolenia od používateľa ohľadne využívania kamery. Dodatočne sme boli požiadaní, aby sme rozšírili dokument o zásadách ochrany osobných údajov, konkrétne časť, ktorá popisuje využívanie dát z kamery. Po splnení požiadaviek bolo publikovanie úspešné [53].

3.3.3 Zásady ochrany osobných údajov

Privacy policy, čiže zásady ochrany osobných údajov, je právny dokument, ktorý popisuje, ako organizácia alebo webová stránka, či aplikácia zhromažďuje, používa, zdieľa a chráni

osobné údaje svojich užívateľov. Obsahuje informácie o tom, aké typy údajov sa zbierajú, na aký účel a ako užívatelia môžu upravovať svoje osobné údaje. Aplikácia využíva dáta z kamery a preto bolo nutné podrobne opísať, ako s nimi pracujeme. Rozšírenú časť o dátach z kamery uvádzame na obrázku 38, kde jasne popisujeme, že dáta zostávajú na zariadení a nie sú zhromažďované. Následne sme uviedli, že využívame TrueDepth kameru a jej dáta, sú hneď transformované na MIDI správy, ktoré pomáhajú vytvoriť mechanizmus, kde stupeň otvorenosti úst ovláda sustain pedál v hudobnom programe. Dáta sú využité v reálnom čase a nie sú uchovávané ani zdieľané s tretími stranami. Kompletný dokument sa nachádza v prílohe C.



Obrázok 38 – Výber zo zásad ochrany osobných údajov

4 Overenie riešenia a jeho iterácie

4.1 Spätná väzba používateľov

Spätná väzba počas fázy po publikovaní MVP je kľúčová, pretože umožňuje vývojárom identifikovať priority pre ďalší vývoj a zlepšiť funkcionality aplikácie podľa potrieb užívateľov už v ranných štádiách produktu. Pre nás bolo kľúčové získať túto spätnú väzbu, a preto sme aktívne komunikovali na rôznych fórach, vrátane Audiobus fóra [54] a taktiež sme získali individuálnu odozvu prostredníctvom e-mailov. Okrem toho sme získali recenziu v internetovom nemeckom magazíne [55], ktorá prilákala viacerých užívateľov, ktorí následne zdieľali svoje názory a skúsenosti v komentároch pod recenziou. Medzi spätnú väzbu sme zahrnuli aj postrehy z interného testovania a analytických nástrojov prostredia Xcode.

4.2 Iterácie riešenia

V tabuľke 4 je možné vidieť prehľad a riešenie spätnej väzby vzhľadom na verziu aplikácie.

Verzia aplikácie	Spätná väzba z predchádzajúcej verzie	Riešenie	Zdroj
1.1	Pri nasledovaní návodu sa po úspešnom pripojení nedozvedám o stave pripojenia, keďže status pripojenia sa nachádza pod momentálne prezentovanou obrazovkou.	Po úspešnom pripojení sa automaticky zatvára prezentovaná obrazovka návodu, keďže používateľ už úlohu z návodu splnil a ďalej návod nepotrebuje.	Interné testovanie
1.1	Z analytických údajov zisťujeme, že aplikáciu si ľudia sťahujú, no nedostávame žiadnu spätnú väzbu.	Vytvorenie obrazovky, ktorá odkazuje na stránku vývojára, emailový kontakt a zanechanie hodnotenia na AppStore.	Analytické nástroje
1.2	Ak nepovolím aplikácii prístup ku kamere, tak aplikácia nejaví známky funkčnosti a na obrazovke nevidíme, čo je problémom.	Zobrazenie hlásenia, ktoré hovorí o nepovolenom prístupe ku kamere. Používateľ má k dispozícii tlačidlo, ktoré ho presmeruje	Interné testovanie

		priamo do systémových nastavení, kde môže prístup ku kamere povoliť.	
1.2	Ak si nainštalujem aplikáciu do zariadenia, ktoré nedisponuje TrueDepth kamerou, tak sa aplikácia javí ako nefunkčná.	Zobrazenie hlásenia, ktoré hovorí o nekompatibilite zariadenia.	Interné testovanie
1.3	Z analytických dát sme zistili, že približne 25% používateľov si stiahlo aplikáciu na iPad.	Prispôsobenie zobrazenia aplikácie pre iPad.	Analytické nástroje
1.4	Nemám počítač Mac. Dala by sa aplikácia pripojiť namiesto toho k iPadu?	Pridanie komunikácie prostredníctvom BLE., pomocou ktorého je možné pripojiť aplikáciu napríklad ku GarageBand na iPade.	Email
1.4	Sledovanie tváre je na mieste, ale chýba podpora pre AUv3 pluginy.	Pridanie podpory načítania AUv3 pluginu, ktorý je možné ovládať pomocou pripojených MIDI klávesov.	Audiobus fórum
1.5	Ak moja tvár nie je snímaná, tak na obrazovke nevidím žiadne upozornenie.	Pridanie upozornenia, ktoré hovorí o tom, že aplikácia momentálne nezaznamenáva žiadnu tvár.	Interné testovanie
1.5	Ak je nedostatočné osvetlenie tváre, tak aplikácia nepracuje správne a o slabom osvetlení aplikácia neposkytuje žiadne upozornenie.	Pridanie upozornenia, ktoré hovorí o tom, že osvetlenie tváre nie je pre správne rozpoznanie dostačujúce.	Interné testovanie
1.5	Popis pre nastavenie prahu „Responsivity“ je mätúci.	Nahradenie popisu za „Threshold“.	Audiobus fórum

Tabuľka 4 – Spätná väzba a jej riešenie

4.3 Výsledok SUS dotazníku

Po publikovaní verzie 1.5 sme uznali za vhodné rozoslať malej vzorke piatich používateľov SUS dotazník, keďže považujeme aplikáciu z hľadiska HCI za vhodne navrhnutú. Výsledok ukázal, že naše riešenie dosahuje hodnotu 79, čo je podobný výsledok, ako pre zariadenie (iPhone)[10], na ktorom je spustená naša aplikácia. Tento výsledok považujeme za pozitívny a naznačuje, že riešenie by malo byť pre používateľov zariadenia iPhone použiteľné.

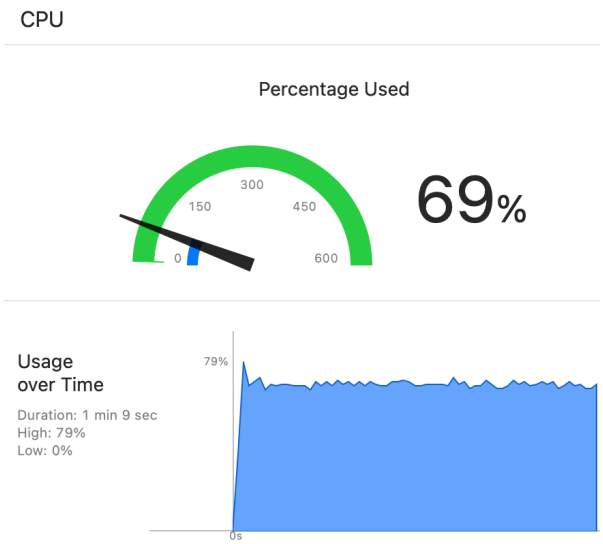
4.4 Záťažové testovanie

4.4.1 Zaťaženie systému

Xcode ponúka robustné testovacie nástroje záťaže, ktoré pomáhajú vývojárom overiť výkon ich aplikácií v rôznych podmienkach. Súčasťou týchto nástrojov je monitorovanie výkonu, ako je spotreba pamäte a výkon procesora.

Zaťaženie procesoru

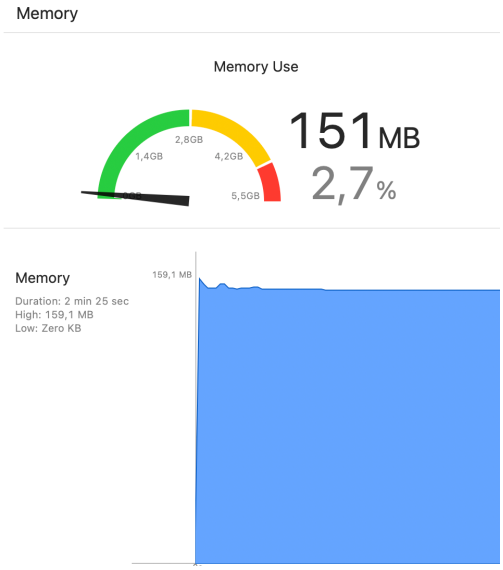
V prostredí Xcode dokážeme v reálnom čase sledovať percentuálne vyťaženie procesoru (graf 1). Prostredie ponúka aj zobrazenie v grafe, ktorý ukazuje zaťaženie procesora v čase. Na základe grafu môžeme pozorovať konštantné zaťaženie, čo potvrdzuje kontinuálne rozpoznávanie tváre. Hodnota zaťaženia je udávaná v percentách, pričom 100% patrí jednému jadru procesora.



Graf 1 – Zaťaženie procesoru

Zaťaženie operačnej pamäte

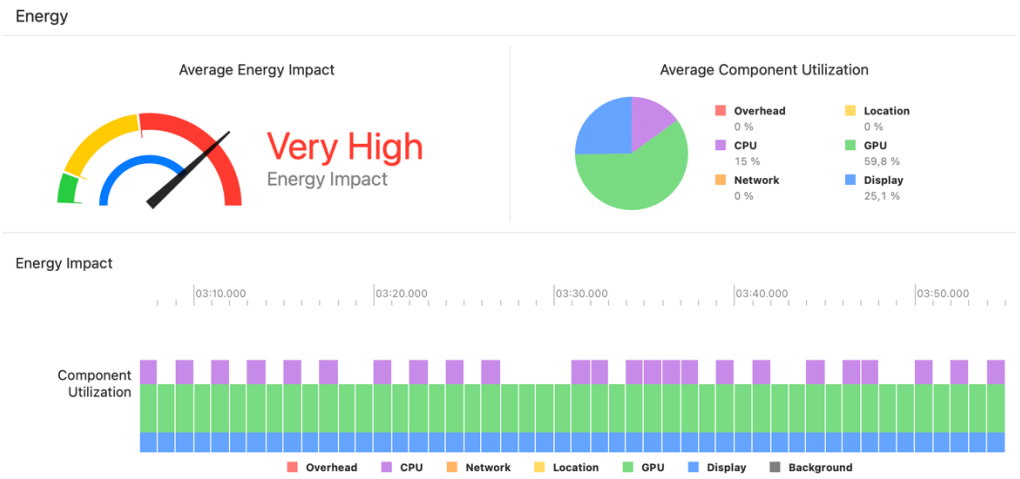
Podobne ako záťaž procesoru, môžeme sledovať aj využitie operačnej pamäte (graf 2). Na základe grafu dokážeme identifikovať konštantné využitie operačnej pamäte, ktoré činí približne 150MB.



Graf 2 – Zaťaženie operačnej pamäte

Spotreba energie

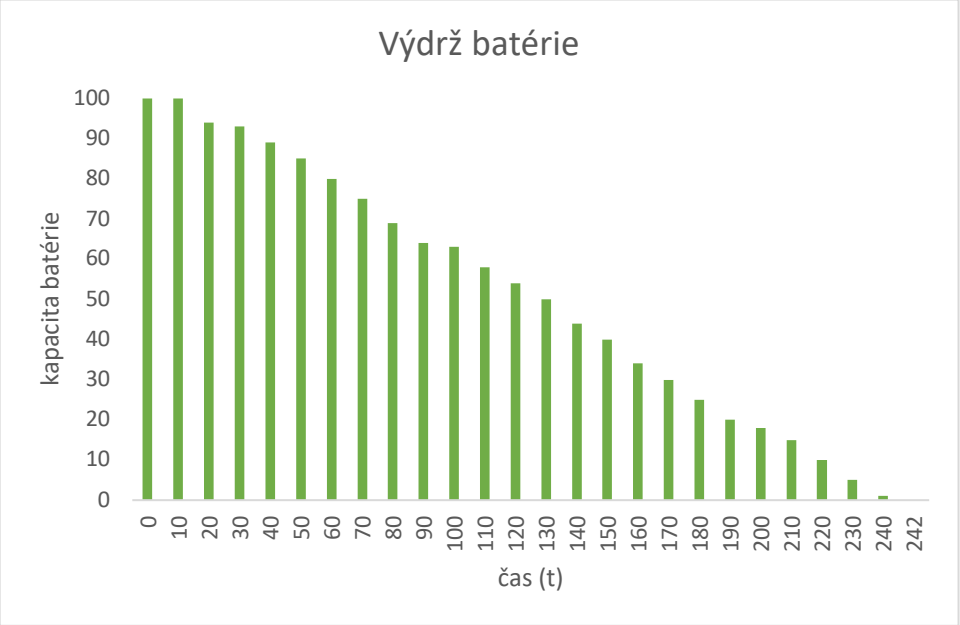
Xcode obsahuje nástroj pre meranie spotreby energie (graf 3), ktorý klasifikuje spotrebu našej aplikácie ako veľmi vysokú. Meranie ukazuje, že najväčšiu spotrebu energie má grafická výpočtová jednotka, ktorá je značne vyťažená réžiou ARKit frameworku. Meranie nám naznačuje, že pri používaní aplikácie, by mohol nastať problém s výdržou batérie zariadenia.



Graf 3 – Spotreba energie

4.4.2 Výdrž batérie

Merania záťaže systému odhalili, že aplikácia značne využíva výpočtové zdroje zariadenia, čo môže výrazne ovplyvniť výdrž batérie. Používateľ sa môže ocitnúť v situácii, kedy chce využiť aplikáciu v prostredí, ktoré nedisponuje elektrickým pripojením. Preto sme navrhli meranie, ktoré ukazuje možný čas výdrže pri plnom nabití zariadenia. Pre merania sme zvolili aktuálny model iPhone 15 [56], ktorý prešiel počiatočnými cyklami nabíjania. To nám zabezpečilo, že sa na pozadí nevykonávali žiadne počiatočné konfigurácie, operácie a optimalizácie na úrovni operačného systému. Ako metódu komunikácie sme zvolili BLE, aby sme obmedzili systémové operácie vyžadujúce internet. Pre meranie stavu batérie sme využili natívne API volanie (obrázok)[57]. Na grafe 4 je možné vidieť progres stavu batérie v čase. Kapacita batérie je udávaná v percentách a čas v minútach. Nameraná doba výdrže batérie sú 4 hodiny a 2 minúty.



Graf 4 – Výdrž energie

Záver

Cieľom našej práce bolo navrhnúť, implementovať a overiť alternatívne riešenie klavírneho pedálu. Analyzovali sme históriu interakcie človek-počítač v osobných počítačoch. Následne sme analyzovali metódy a metriky, ktoré sú využívané pri vývoji HCI. Popísali sme prístup spoločnosti Apple vzhľadom k prístupnosti. Opísali sme zaujímavé hudobné aplikácie, ktoré využívajú rozšírenú realitu. Preskúmali sme existujúce riešenia alternatívneho klavírneho pedálu, vďaka čomu sme identifikovali kľúčové vlastnosti interakcie, na základe ktorých sme riešenia medzi sebou porovnali. Táto analýza nám poskytla hlboký pohľad do problematiky, ktorý nám pomohol pri návrhu riešenia.

Navrhli sme experimentálny nenákladný prototyp, ktorý ukázal, že vhodným gestom pre ovládanie pedálu je pootvorenie úst. Navrhli a implementovali sme samotnú interakciu a komunikáciu medzi aplikáciou a hudobným nástrojom. Pre implementáciu interakcie sme uznali metódu merania vzdialenosti medzi hornou a spodnou perou pomocou knižnice ARKit ako vhodnú.

Aplikáciu sme publikovali ako MVP, čo nám umožnilo získať spätnú väzbu v rannej fáze vývoja. Spätnú väzbu sme zapracovali do riešenia v podobe aktualizácií. Výsledok SUS dotazníku nám ukázal, že použiteľnosť aplikácia je akceptovateľná. Pomocou nástrojov prostredia Xcode sme merali zaťaženie systému, ktoré odhalilo vysokú spotrebu energie, čo viedlo k nášmu meraniu výdrže batérie. Výdrž batérie pri spustenej aplikácii je dostačujúca pre rôzne príležitosti, ako domáci nácvik alebo koncert.

Aplikáciu je možné stiahnuť a inštalovať z internetového obchodu Apple AppStore, kde je bez poplatku dostupná po celom svete.

V súčasnosti experimentujeme s návrhom a implementáciou ďalšej metódy komunikácie prostredníctvom MicroBitu [58], ku ktorému je cez BLE pripojená aplikácia a doska je zapojená pomocou 6.3mm jack konektoru do klavíru, kde po prijatí správy z aplikácie spína kontakt, čím aktivuje pedál. Očakávame podporu knižnice pre rozpoznávanie tváre používateľov headsetu Apple Vision Pro, pre ktorý by sme chceli vytvoriť špeciálnu verziu, ktorá by využívala možnosti priestorového počítania. V budúcich verziách aplikácie sa budeme venovať integrácii funkcionalít systémovej prístupnosti, čo by pomohlo začleniť používateľov s viacerými zdravotnými postihnutiami.

Zoznam bibliografických odkazov

[1] Interaction Design, *What is Human-Computer Interaction (HCI),* [cit. 1.5.2024]. Dostupné z: https://www.interaction-design.org/literature/topics/human-computer-interaction

[2] WINOGRAD, Terry. *Lecture 11 - HCI History,* [cit. 1.5.2024]. Dostupné z: https://hci.stanford.edu/courses/cs147/2006/slides/11-history/cs147-history.pdf

[3] Cult of Mac, *Apple-I starts a revolution*, [cit. 1.5.2024]. Dostupné z: https://www.cultofmac.com/475761/apple-1-launch/?utm_content=cmp-true

[4] The Interface Experience, *Apple II, 1977* , [cit. 1.5.2024]. Dostupné z: https://interface-experience.org/objects/apple-ii/

[5] Techbox, *Apple pred 37 rokmi predstavil svoj prvý Macintosh*, [cit. 1.5.2024]. Dostupné z: https://www.techbox.sk/apple-pred-37-rokmi-predstavil-svoj-prvy-macintosh

[6] Wikipedia, *Powerbook 100*, [cit. 1.5.2024]. Dostupné z: https://sk.m.wikipedia.org/wiki/Súbor:Powerbook_100_pose.jpg

[7] Businessinsider, *Watch Steve Jobs unveil the very first iPhone 10 years ago*, [cit. 1.5.2024]. Dostupné z: https://www.businessinsider.in/tech/watch-steve-jobs-unveil-the-very-first-iphone-10-years-ago-today/slidelist/56421808.cms

[8] Apple Inc., *Apple Vision Pro available in the US on february 2*, [cit. 1.5.2024]. Dostupné z: https://www.apple.com/newsroom/2024/01/apple-vision-pro-available-in-the-us-on-february-2/

[9] Zain Books, *Goals and evolution of HCI*, [cit. 1.5.2024]. Dostupné z: http://www.zainbooks.com/books/computer-sciences/human-computer-interaction_4_goals-and-evolution-of-human-computer-interaction.html

[10] Zain Books, *Design principles*, [cit. 1.5.2024]. Dostupné z: http://www.zainbooks.com/books/computer-sciences/human-computerinteraction_12_design-principles.html

[11] JEŽÍKOVÁ, Kristína. *System usability scale: meradlo použiteľnosti* [cit. 1.5.2024]. Dostupné z: https://medium.com/design-kisk/system-usability-scale-meradlo-použiteľnosti-6e2a95f8b73f

[12] SeeWriteHear, *What is Accessibility?,* [cit. 1.5.2024]. Dostupné z: https://www.seewritehear.com/learn/what-is-accessibility/

[13] Apple Inc., *Accessibility*, [cit. 1.5.2024]. Dostupné z: https://www.apple.com/accessibility/

[14] Apple Inc., *GarageBand*, [cit. 1.5.2024]. Dostupné z: https://www.apple.com/ios/garageband/

[15] Apple Inc., *MIDI Mouth Controller*, [cit. 1.5.2024]. Dostupné z: https://apps.apple.com/sk/app/midi-mouth-controller/id1447471921?l=sk

[16] Apple Inc., *Musikraken*, [cit. 1.5.2024]. Dostupné z: https://apps.apple.com/sk/app/musikraken/id1538781007?l=sk

[17] Steingraeber, *Pedal devices for pianists in wheelchair*, [cit. 1.5.2024]. Dostupné z: https://www.steingraeber.de/en/innovationen/pedal-devices-for-pianists-in-wheelchairs/

[18] VUT, *A pianist in a wheelchair can also use pedals to play thanks to a unique device it took the researchers from the mechatronic laboratory half a year to develop the prototype*, [cit. 1.5.2024]. Dostupné z: https://www.vut.cz/en/but/news-f19528/a-pianist-in-a-wheelchair-can-also-use-pedals-to-play-thanks-to-a-unique-device-it-took-the-researchers-from-the-mechatronic-laboratory-half-a-year-to-develop-the-prototype-d193890?aid_redir=1

[19] Apple Inc., *Vision*, [cit. 1.5.2024]. Dostupné z: https://developer.apple.com/documentation/vision/

[20] Apple Inc.,, *Tracking and Visualizing Faces*, [cit. 1.5.2024]. Dostupné z: https://developer.apple.com/documentation/arkit/arkit_in_ios/content_anchors/tracking_and_visualizing_faces

[21] Apple Inc., *blendShapes*, [cit. 1.5.2024]. Dostupné z: https://developer.apple.com/documentation/arkit/arfaceanchor/2928251-blendshapes

[22] Stackoverflow*, How can I get the points in face* mesh, [cit. 1.5.2024]. Dostupné z: https://stackoverflow.com/questions/53928038/how-can-i-get-the-points-in-face-mesh-like-eye-eyebrow-lip-mouth-nose-using

[23] Apple Inc., Zdieľanie informácií MIDI prostredníctvom siete v apke Nastavenia MIDI audia na Macu, [cit. 1.5.2024]. Dostupné z: https://support.apple.com/sk-sk/guide/audio-midi-setup/ams1012/mac

[24] Apple Inc., *MIDI Networking*, [cit. 1.5.2024]. Dostupné z: https://developer.apple.com/documentation/coremidi/midi_networking

[25] Apple Inc., *Používanie Bluetooth MIDI zariadenia s dotykovými nástrojmi v GarageBande*, [cit. 1.5.2024]. Dostupné z: https://support.apple.com/sk-sk/guide/garageband-iphone/chse356a0321/ios

[26] Apple Inc., *CoreBluetooth*, [cit. 1.5.2024]. Dostupné z: https://developer.apple.com/documentation/corebluetooth

[27] Apple Inc., *Audio Unit v3 Plug-Ins*, [cit. 1.5.2024]. Dostupné z: https://developer.apple.com/documentation/audiotoolbox/audio_unit_v3_plug-ins
[28] Audiokit, [cit. 2.4.2024]. Dostupné z: https://github.com/AudioKit
[29] Snoize, *MIDIMonitor* [cit. 1.5.2024]. Dostupné z: https://www.snoize.com/MIDIMonitor/
[30] Apple Inc., *Human interface guidelines*, [cit. 1.5.2024]. Dostupné z: https://developer.apple.com/design/human-interface-guidelines
[31] Apple Inc., *SwiftUI*, [cit. 1.5.2024]. Dostupné z: https://developer.apple.com/documentation/swiftui/
[32] Apple Inc., *SF Symbols 5*, [cit. 1.5.2024]. Dostupné z: https://developer.apple.com/sf-symbols/
[33] Apple Inc., *idleTimerDisabled*, [cit. 1.5.2024]. Dostupné z: https://developer.apple.com/documentation/uikit/uiapplication/1623070-idletimerdisabled
[34] Apple Inc., *simd_distance(_:_:)*, [cit. 1.5.2024]. Dostupné z: https://developer.apple.com/documentation/accelerate/2917765-simd_distance
[35] Apple Inc., *MIDIClientCreate(_:_:_:_:)*, [cit. 1.5.2024]. Dostupné z: https://developer.apple.com/documentation/coremidi/1495360-midiclientcreate
[36] Apple Inc., *MIDIOutputPortCreate(_:_:_:)*, [cit. 1.5.2024]. Dostupné z: https://developer.apple.com/documentation/coremidi/1495166-midioutputportcreate
[37] Apple Inc., *MIDINetworkSession,* [cit. 1.5.2024]. Dostupné z: https://developer.apple.com/documentation/coremidi/midinetworksession?language=objc
[38] Apple Inc., *MIDIPacketListAdd,* [cit. 1.5.2024]. Dostupné z: https://developer.apple.com/documentation/coremidi/1495128-midipacketlistadd?language=objc
[39] Apple Inc., *MIDISend(_:_:_:),* [cit. 1.5.2024]. Dostupné z: https://developer.apple.com/documentation/coremidi/1495289-midisend
[40] Apple Inc., *CABTMIDILocalPeripheralViewController,* [cit. 1.5.2024]. Dostupné z: https://developer.apple.com/documentation/coreaudiokit/cabtmidilocalperipheralviewcontroller?changes=_3_9
[41] Apple Inc., *CBMutableCharacteristic,* [cit. 1.5.2024]. Dostupné z: https://developer.apple.com/documentation/corebluetooth/cbmutablecharacteristic
[42] Apple Inc., *CBPeripheralManager,* [cit. 1.5.2024]. Dostupné z: https://developer.apple.com/documentation/corebluetooth/cbperipheralmanager
[43] Apple Inc., *startAdvertising(_:),* [cit. 1.5.2024]. Dostupné z:

https://developer.apple.com/documentation/corebluetooth/cbperipheralmanager/startadvertising(_:)
[44] Apple Inc., *AVAudioUnitComponentManager,* [cit. 1.5.2024]. Dostupné z: https://developer.apple.com/documentation/avfaudio/avaudiounitcomponentmanager
[45] Apple Inc., *Audio Unit Types,* [cit. 1.5.2024]. Dostupné : https://developer.apple.com/documentation/audiotoolbox/1584142-audio_unit_types
[46] Apple Inc., *mainMixerNode,* [cit. 1.5.2024]. Dostupné z: https://developer.apple.com/documentation/avfaudio/avaudioengine/1385813-mainmixernode
[47] Apple Inc., *AVAudioEngine,* [cit. 1.5.2024]. Dostupné z: https://developer.apple.com/documentation/avfaudio/avaudioengine
[48] Apple Inc., *AUAudioUnitPreset,* [cit. 1.5.2024]. Dostupné z: https://developer.apple.com/documentation/audiotoolbox/auaudiounitpreset
[49] Audiokit, *MIDIListener,* [cit. 1.5.2024]. Dostupné z: https://www.audiokit.io/AudioKit/documentation/audiokit/midilistener
[50] Magika Studio, *Deciding between MVP and full product launch for your business,* [cit. 1.5.2024]. Dostupné z: https://www.magika.studio/post/deciding-between-mvp-and-full-product-launch-for-your-business
[51] Apple Inc., *AppStore,* [cit. 1.5.2024]. Dostupné z: https://www.apple.com/app-store/
[52] Apple Inc., *AppStore Connect,* [cit. 1.5.2024]. Dostupné z: https://developer.apple.com/app-store-connect/
[53] Apple Inc., *MouthPedal,* [cit. 1.5.2024]. Dostupné z: https://apps.apple.com/sk/app/mouthpedal/id6448126913?l=sk
[54] Audiobus, *Forum,* [cit. 1.5.2024]. Dostupné z: https://forum.audiob.us
[55] iPhone ticker, *MouthPedal app MIDI steuerung per face id und mundeoffung,* [cit. 1.5.2024]. Dostupné z: https://www.iphone-ticker.de/mouthpedal-app-midi-steuerung-per-face-id-und-mundoeffnung-219287/
[56] Apple Inc., *iPhone 15,* [cit. 1.5.2024]. Dostupné z: https://www.apple.com/sk/iphone-15/
[57] Apple Inc., v*batteryLevel,* [cit. 1.5.2024]. Dostupné z: https://developer.apple.com/documentation/uikit/uidevice/1620042-batterylevel
[58] Microsoft, Microbit, [cit. 1.5.2024]. Dostupné z: https://microbit.org

Zoznam príloh

- Príloha A – Návod pre pripojenie aplikácie ku Mac počítaču pomocou MIDI Network
- Príloha B – Návod pre pripojenie aplikácie ku tabletu iPad pomocou BLE
- Píloha C – Zásady ochrany osobných údajov
- Príloha D – Praktická časť
    - Xcode_project_folder
    - Návod_na_spustenie_projektu.pdf

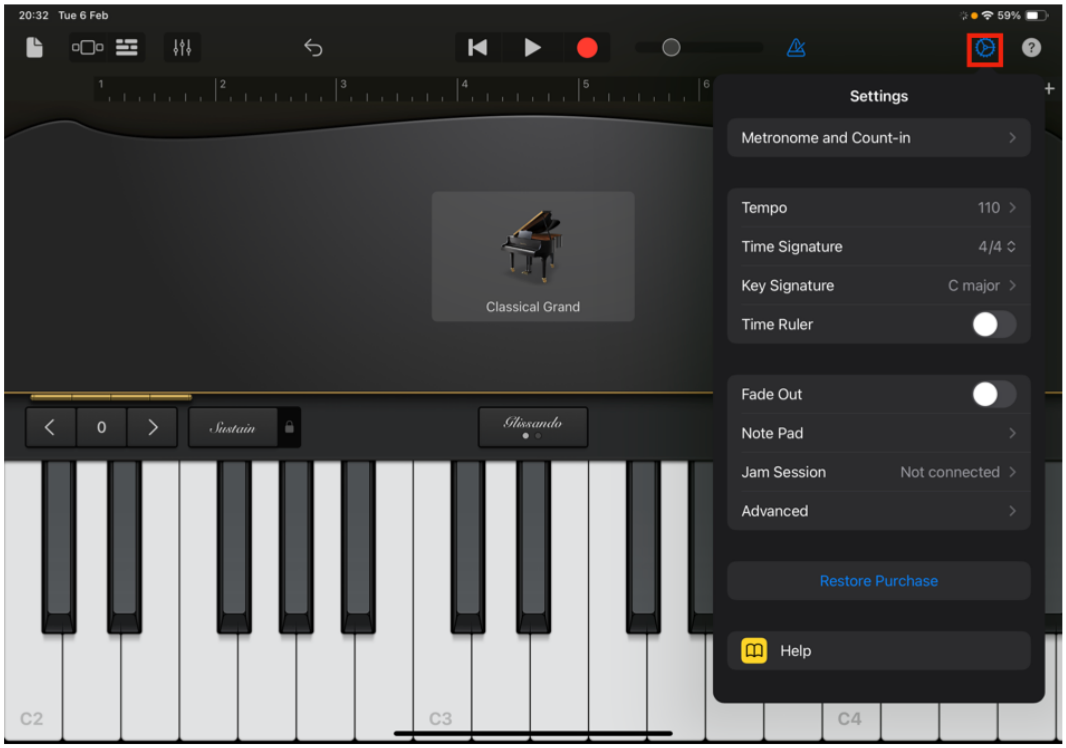
Príloha A

Step by step tutorial

Connecting to iPad using MIDI over BLE

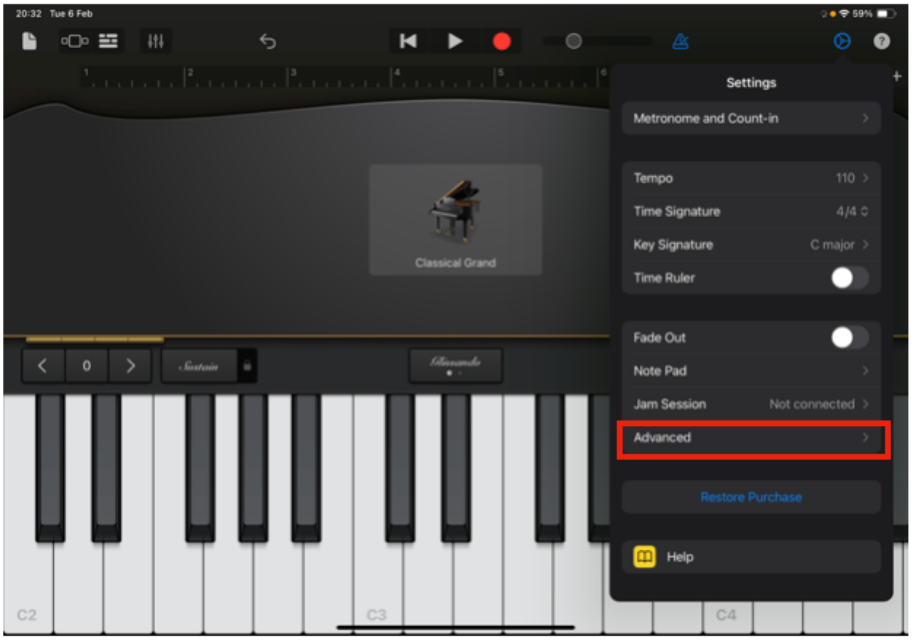
Step 1

Go to GarageBand “Settings”



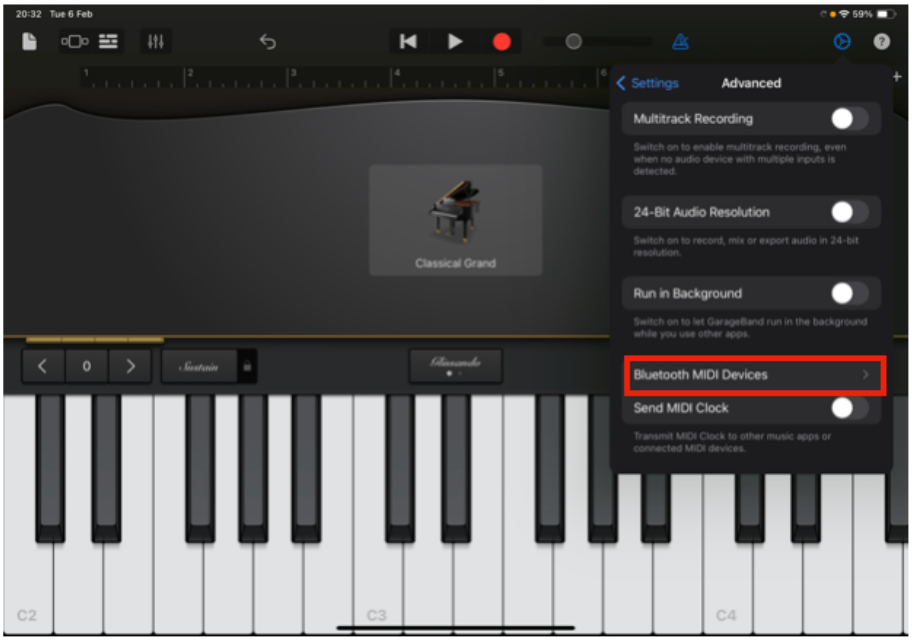
Step 2

Go to “Advanced”



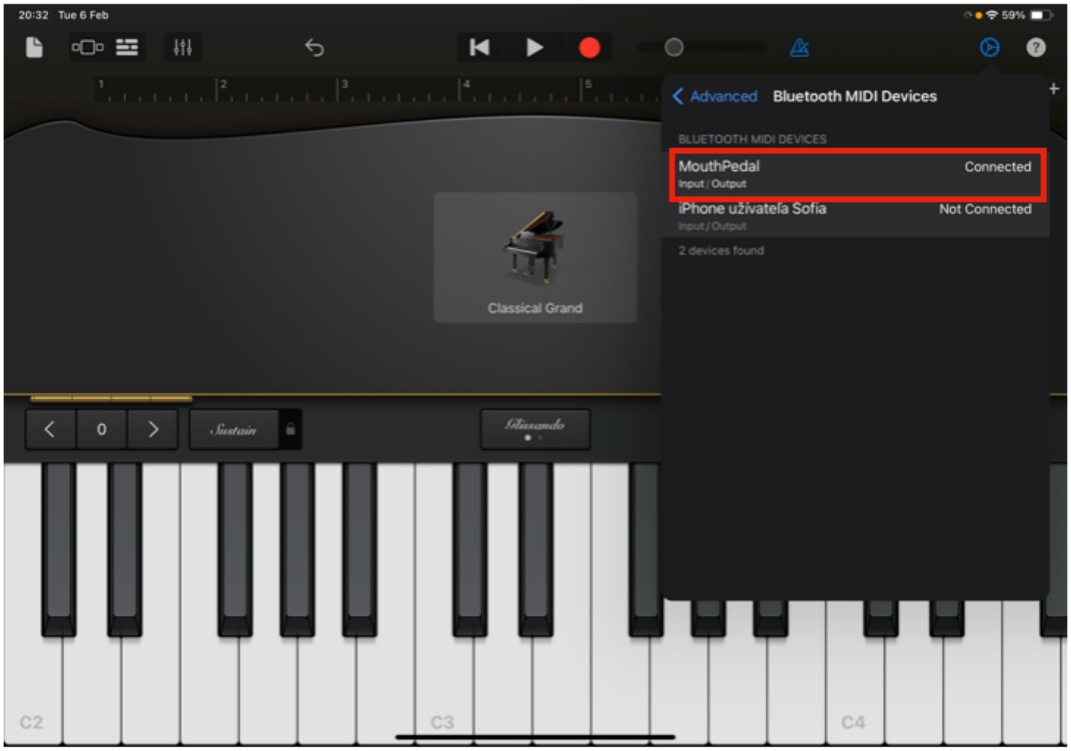
Step 3

Go to “Bluetooth MIDI Devices”



Step 4

Connect to “MouthPedal”



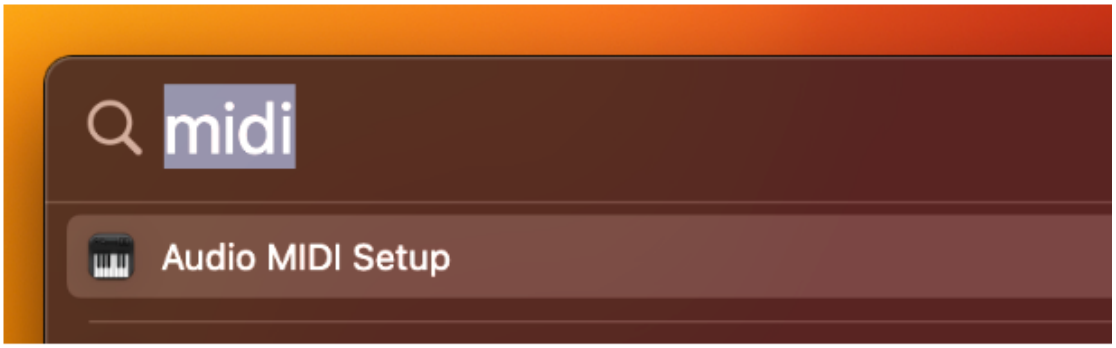
Príloha B

Step by step tutorial

Connecting to Mac

Step 1

Launch “Audio MIDI Setup”



Step 2

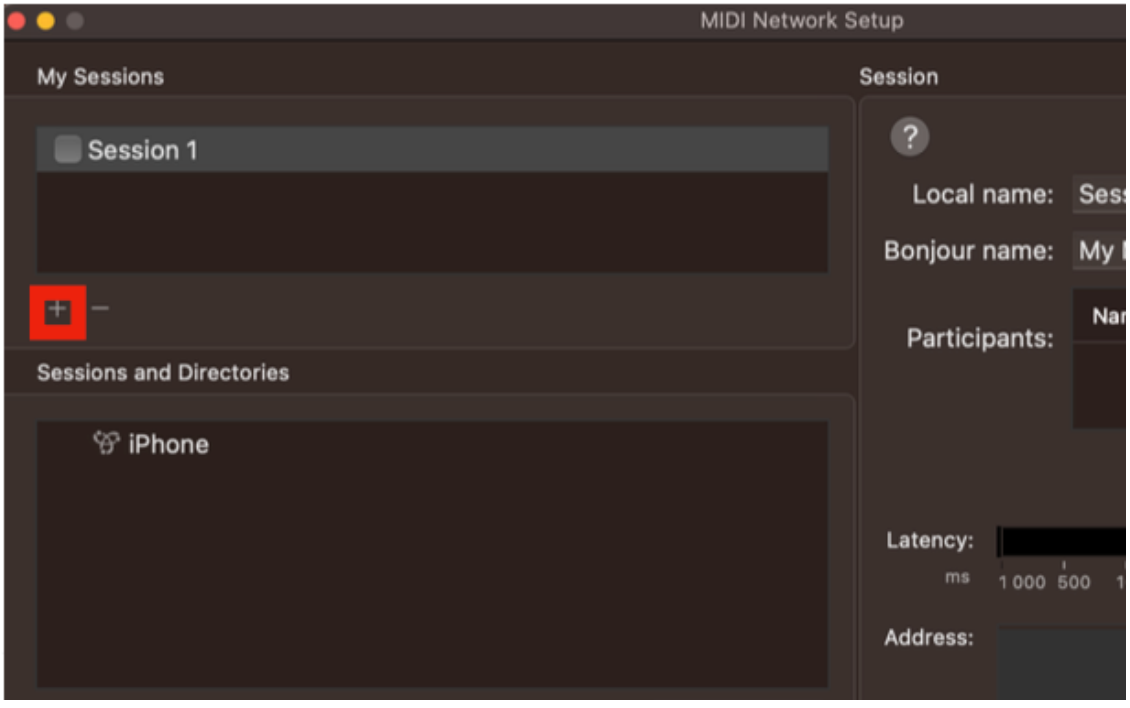
Window -> Show MIDI Studio

Step 3

MIDI Studio -> Open MIDI Network Setup

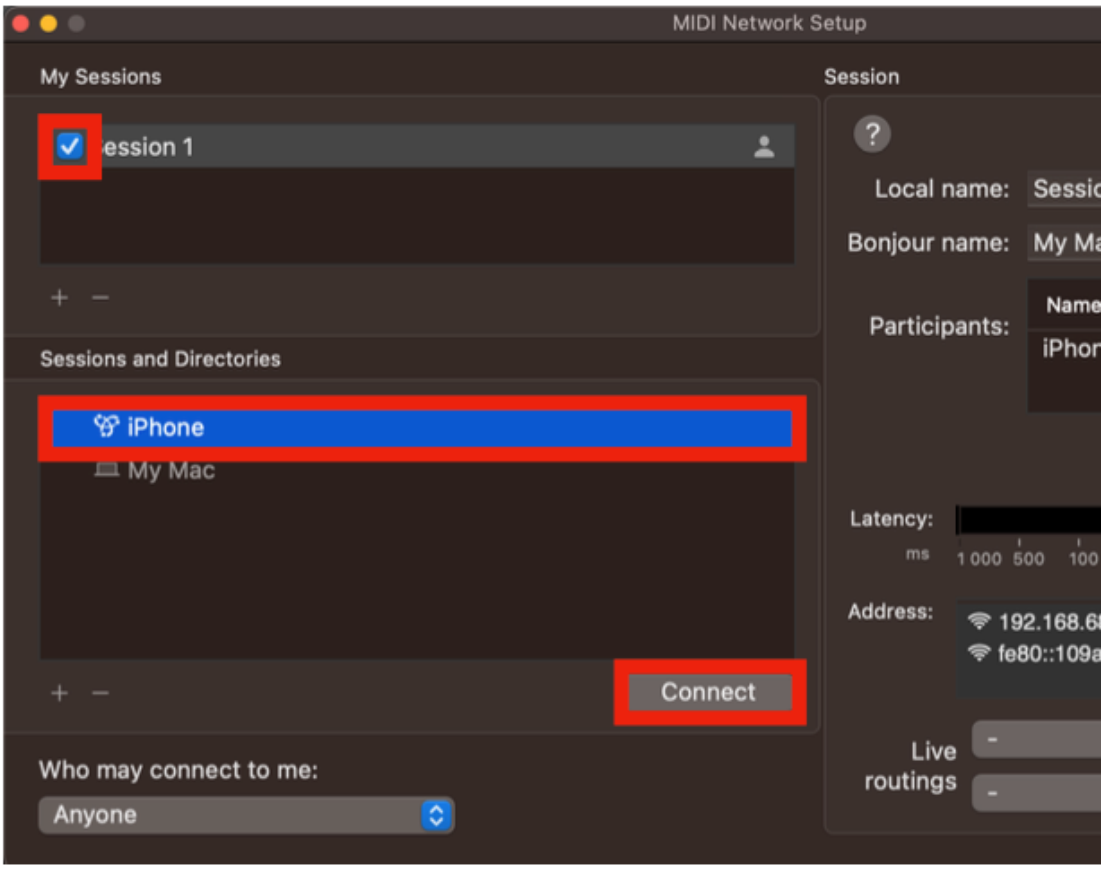
Step 4

Click “+” sign to add Session



Step 5

Enable session



Príloha C

Privacy Policy

Peter Hafner built the MouthPedal app as a Free app. This SERVICE is provided by Peter Hafner at no cost and is intended for use as is.

This page is used to inform visitors regarding my policies with the collection, use, and disclosure of Personal Information if anyone decided to use my Service.

If you choose to use my Service, then you agree to the collection and use of information in relation to this policy. The Personal Information that I collect is used for providing and improving the Service. I will not use or share your information with anyone except as described in this Privacy Policy.

The terms used in this Privacy Policy have the same meanings as in our Terms and Conditions, which are accessible at MouthPedal unless otherwise defined in this Privacy Policy.

Information Collection and Use

For a better experience, while using our Service, I may require you to provide us with certain personally identifiable information. The information that I request will be retained on your device and is not collected by me in any way. Application utilizes Apple's TrueDepth API to access data related to the degree of mouth openness. This specific data is immediately transformed into MIDI messages to provide a control mechanism, where the degree of mouth openness can control sustain pedal of DAW on Mac, and it is not used for any other purpose. The data captured is processed in real time and not stored, thereby leaving no long-term records. We do not share this data with any third parties.

Log Data

I want to inform you that whenever you use my Service, in a case of an error in the app I collect data and information (through third-party products) on your phone called Log Data. This Log Data may include information such as your device Internet Protocol (“IP”) address, device name, operating system version, the configuration of the app when utilizing my Service, the time and date of your use of the Service, and other statistics.

Cookies

Cookies are files with a small amount of data that are commonly used as anonymous unique identifiers. These are sent to your browser from the websites that you visit and are stored on your device's internal memory.

This Service does not use these “cookies” explicitly. However, the app may use third-party code and libraries that use “cookies” to collect information and improve their services. You have the option to either accept or refuse these cookies and know when a cookie is being sent to your device. If you choose to refuse our cookies, you may not be able to use some portions of this Service.

Service Providers

I may employ third-party companies and individuals due to the following reasons:

- To facilitate our Service;
- To provide the Service on our behalf;
- To perform Service-related services; or
- To assist us in analyzing how our Service is used.

I want to inform users of this Service that these third parties have access to their Personal Information. The reason is to perform the tasks assigned to them on our behalf. However, they are obligated not to disclose or use the information for any other purpose.

Security

I value your trust in providing us your Personal Information, thus we are striving to use commercially acceptable means of protecting it. But remember that no method of transmission over the internet, or method of electronic storage is 100% secure and reliable, and I cannot guarantee its absolute security.

Links to Other Sites

This Service may contain links to other sites. If you click on a third-party link, you will be directed to that site. Note that these external sites are not operated by me. Therefore, I strongly advise you to review the Privacy Policy of these websites. I have no control over and assume no responsibility for the content, privacy policies, or practices of any third-party sites or services.

Links to Other Sites

This Service may contain links to other sites. If you click on a third-party link, you will be directed to that site. Note that these external sites are not operated by me. Therefore, I strongly advise you to review the Privacy Policy of these websites. I have no control over and assume no responsibility for the content, privacy policies, or practices of any third-party sites or services.

Children's Privacy

I do not knowingly collect personally identifiable information from children. I encourage all children to never submit any personally identifiable information through the Application and/or Services. I encourage parents and legal guardians to monitor their children's Internet usage and to help enforce this Policy by instructing their children never to provide personally identifiable information through the Application and/or Services without their permission. If you have reason to believe that a child has provided personally identifiable information to us through the Application and/or Services, please contact us. You must also be at least 16 years of age to consent to the processing of your personally identifiable information in your country (in some countries we may allow your parent or guardian to do so on your behalf).

Changes to This Privacy Policy

I may update our Privacy Policy from time to time. Thus, you are advised to review this page periodically for any changes. I will notify you of any changes by posting the new Privacy Policy on this page.

This policy is effective as of 2022-06-7

Contact Us

If you have any questions or suggestions about my Privacy Policy, do not hesitate to contact me at hafner.developer@gmail.com.

This privacy policy page was created at privacypolicytemplate.net and modified/generated by App Privacy Policy Generator