TALLINNA TEHNIKAÜLIKOOL
Infotehnoloogia teaduskond

Richard Bossenko IAPB155417

LAULMISE MOBIILIRAKENDUS ANDROIDI PLATVORMIL: STARME

Bakalaureuse töö

Juhendaja: Gert Kanter

Tehnikateaduse
Magistrikraad

Tallinn 2019

Autorideklaratsioon

Kinnitan, et olen koostanud antud lõputöö iseseisvalt ning seda ei ole kellegi teise poolt varem kaitsmisele esitatud. Kõik töö koostamisel kasutatud teiste autorite tööd, olulised seisukohad, kirjandusallikatest ja mujalt pärinevad andmed on töös viidatud.

Autor: Richard Bossenko

29.04.2019

Annotatsioon

Käesoleva bakalaureusetöö raames on tehtud Androidi mobiilirakendus, mis on laulmise ning sotsiaalmeedia platvorm. Rakendus on mõeldud inimestele, kes oskavad laulda ning otsivad keskkonda kus enda laulmisoskusi teistega jagada või neile, kes ei oska laulda, kuid soovivad seda teha ja ei tea kust alustada.

Rakenduse laulmise vaates kuvatakse kasutajale graafikuna joon mille järgi tuleb laulda, kontrollitakse pidevalt kasutaja häälekõrgust ning antakse kasutajale tagasisidet kas ta laulab valitud laulu antud hetkel õigesti. Peale selle kuvatakse ka laulvale kasutajale laulusõnad.

Rakenduse sotsiaalmeedia pooles saavad kasutajad filmida ja lindistada ennast laulmas, valminud videot jagada teiste kasutajatega, vaadata teiste kasutajate lisatud videoid ning neid videoid laikida ja kommenteerida.

Bakalaureusetöö kirjutamise hetkel on see rakendus valmis ning saadaval Androidi Google Play poes nimega StarMe.

Lõputöö on kirjutatud eesti keeles ning sisaldab teksti 56 lehekülge, 4 peatükki, 16 joonist, 0 tabelit.

Abstract

Singing mobile application for Android: StarMe

This bachelor thesis reviews an Android mobile application. The application is both a social media platform and a singing game. This application is made for people who know how to sing and share their singing results with other people or for people who do not know singing yet and need a place to learn singing.

In the singing section of the application, the user is shown a graph by which the song should be sang, the user's pitch is continuously checked and the user is given feedback on whether or not he is singing the selected song correctly at given time. In addition, lyrics are also shown to the singing user.

In the social media section of the app, users can record themselves singing or they can share the finished video with other users. Additionally they can watch videos from other users, and like and comment their videos.

This bachelor thesis mainly focus on the singing section of the application and only a little is about the application as a whole. Applications development started in 15.04.2018 and author spent at least 160 hours a month on this thesis. The application is finished and is available on Android's Google Play Store.

The thesis is in Estonian and contains 56 pages of text, 4 chapters, 56 figures, 0 tables.

Lühendite ja mõistete sõnastik

Android	Android on tarkvarakomplekt elektroonikaseadmetele, põhiliselt nutitelefonidele. Android hõlmab operatsioonisüsteemi, peamisi rakendusi ja vahetarkvara.
Tagarakendus	Teenuse ülesandeks on vahetult suhelda eesrakendusega. Tagarakenduse ülesandeks on toetada eesrakendust vajalike andmetega kuna tema on lähemal vajalikule ressursile.
Java	Java on objektorienteeritud programmeerimiskeel.
XML	XML on laiendatav märgistuskeel, mis loodi HTMLi asemele kuna HTML oli mõningate ülesannete jaoks liiga piiratud. Antud kontekstis kasutatakse XMLi Androidi rakenduses kasutajaliidese koostamiseks.
JavaScript	JavaScript on skriptikeel, mis suudab suhelda HTML-keeles keeles kirjutatud lähtekoodiga.
AngularJS	AngularJS on JavaScriptil põhinev veebirakenduste arendamiseks tehtud raamistik. AngularJS on loodud selleks, et lihtsustada veebirakenduste arendust.
Neo4j	Neo4j on graafiline andmebaasihaldussüsteem. Päringute keeleks on Neo4j poolt tehtud *Cypher Query Language*.
IOS	On Apple poolt tehtud operatsioonisüsteem, mis on tehtud Apple mobiiltelefonidele.
SDK	SDK on tarkvaraarenduse tööriistade kogumik. See aitab luua tarkvara kindla tarkvara- või riistavaraplatvormi jaoks. Antud töös kasutatakse Android SDK.
Android Studio	Android Studio on tarkvaraplatvorm, mis aitab kasutajal luua Androidi seadmetele rakendusi.
API	API on rakendusliides, mis võimaldab kolmandal osapoolel kasutada juba valmis tehtud lahendusi. APIsid saab importida enda rakendusse ning muuta seda vastavalt enda vajadusele.
WAV	WAV on tihendamata vormiga helifailivorming. WAV on toores heli vorming ning sisaldab endas ka diskreetimissagedust ja bitisügavust. Selle tõttu on WAV failid tavaliselt vägagi mahukad.
MP3	MP3 on helifailivorming, mis on tänu tihendamise tarkvara kasutusele oma mahult oluliselt väiksem kui näiteks toores heli vorminguga WAV.

MP4	MP4 on konteineriformaat MPEG-4st, mis on video- ja audioandmete kadudega pakkimine. MP4 sisaldab erinevalt MP3st nii audio kui ka videofaile.
ViewPort	ViewPort on see mida meie silm korraga näeb, ehk siis seda kutsutakse ka kaameraruumiks. Kaameraruum koosneb koordinaatidest ning neid muutes muutub mida on hetkel näha.
SSH	SSH on võrguprotokoll turvaliseks võrguteenuse opereerimiseks. Enamasti kasutatakse seda kaugetesse arvutisüsteemidesse sisse logimisel.
C++	C++ on multifunktsionaalne programmeerimiskeel, mis toetab muu hulgas ka objektorienteeritud ja üldistavat programmeerimis.
FFT	FFT on Fourier' teisendus on signaalitöötluse fundamentaalne mõiste.
Laik	Tegusõna, mis pärineb ingliskeelsest sõnast *like*. Antud töö kontekstis tähendab tegusõna laikima kellegi teise video all olevale nupule vajutamine. Selle tagajärjel näeb läigitud video kasutaja, et oled tema videot laikinud.
Järgima	On antud rakenduse kontekstis kellegi teise kasutaja profiilil silma peal hoidmine. Ehk kasutaja näeb kui järgitav kasutaja postitab uue video või teeb mõne muu tegevuse rakenduses

Sisukord

Jooniste loetelu

1 Sissejuhatus

Tänapäeval on väga populaarseks tegevuseks laulmine. Üha rohkem ja rohkem inimesi proovib tegeleda laulmisega, kas professionaalselt või üksi kodus. Teisi kuulates tekib inimestel soov samamoodi laulda, kuid ei ole kuskilt laulmist õppida ja ei julgeta võtta endale lauluõpetajat. Antud rakendus on mõeldud nii uutele inimestele, kes soovivad laulmist õppida kui ka lauljatele, kes soovivad viibida keskkonnas, mis on tehtud just lauljate jaoks.

Käesoleva bakalaureusetöö idee tuli autori tolleaegselt ülemuselt, Arnet Mahmastolt, kes pakkus autorile välja antud projekti. Arnet sai omakorda idee Tom Veasey'lt. Ideeks oli Androidi rakendus, mis lubaks kasutajal valida loo mida laulda ning lindistada ennast seda lugu laulmas. Rakendus aitab kasutajal laulda, andes talle ette õiged kõrgused kust laulda, kontrollib mis kõrgusel kasutaja antud hetkel laulab ja annab kasutajale märku kui tegemist on vale kõrgusega. Lisaks näitab rakendus kasutajale laulusõnu, mille järgi laulda. Antud projekt tundus alguses väga ambitsioonikas ning autor ega Arnet ei olnud kindlad, kas sellise rakenduse realiseerimine on võimalik.

Projekti esimene samm oli kirjutada väiksem rakendus, et proovida, kas antud lahendus on võimalik ja kui täpseks seda teha annab. Järgmiseks sammuks oli integreerida see lahendus juba valmisolevasse rakendusse *Picogram*, mille lähtekoodi ostis Arnet, ettevõttelt nimega *Appscript*. Kui see oli tehtud, tuli ümber kirjutada ka nende andmebaasid, muuta kogu rakenduse välimust ning pidevalt täiustada rakenduse põhifookuses olevat laulmise osa. Projekti ning ka rakenduse nimeks sai StarMe.

Selles bakalaureuse töös keskendub autor kõige rohkem rakenduse kolmandale vaatele, milleks on laulmise osa. Kogu projekti kirja panemine oleks liiga mahukas ja laulmise osa tegemine oli autori jaoks kõige suurem ning raskem saavutus.

Lõputöö teises peatükis annab autor ülevaate rakendusest *Picogram*, kirjeldab ka sarnaseid lahendusi ja antud rakenduse konkurenti, juttu on ka autori rollist antud töös. Kolmandas peatükis räägitakse funktsionaalsetest nõuetest, neljas peatükk keskendub

testrakenduse realiseerimisele ning viimane peatükk annab ülevaate kuidas autor ühendas testrakenduse ning *Picogrami* ja mida ning kuidas ta seal muutis.

2 Taust

On mitmeid eri mooduseid laulmist õppida ning jagada ennast laulmas teiste inimestega, kuid see ei ole alati just kõige mugavam ning on vähe rakendusi, mis keskenduvad ainult nendele tegevustele. Sellest tuli ka idee teha rakendus StarMe.

Siinkohal mainib autor, et tegu on litsentseeritud rakendusega. Soovi korral koodi näha või sellega lähemalt tutvuda, tuleb kontakteeruda autoriga. Rakendus on kättesaadaval *Google Play* poes.

2.1 Idee sünd

Rakenduse idee ja suurem osa rahastusest tuli lauljalt ja näitlejalt nimega Tom Veasey, kes on pärit Inglismaalt. Tom on näidelnud tuntud sarjades, kuid antud hetkel on ta lauluõpetaja ning hääletreener. Tema tahtis teha lauljate jaoks eraldi platvormi, kus kõik lauljad saaksid üles laadida oma lindistatud lugusid ning üksteist laulmises aidata.

Kuigi idee tuli Tom'i poolt, siis kõige suurem sõnaõigus rakenduse funktsionaalsuses ja välimuses oli autoril. Tom ja autor suhtlesid ning arutasid paljud detailid läbi enne nende teostamist.

Tom alustas ligikaudu 2019 aastal veebruaris oma hääletreenimisvideotega StarMe nime alt ja jagas neid StarMe veebilehel [1] ning StarMe *Instagramis* [2].

2.2 Picogram

Picogram [3] on sotsiaalmeedia Androidi rakendus, kus on 5 põhilist vaadet. Ning need jagunevad järgmiselt:

- Esimeses vaates on kõikide kasutajate, keda sa jälgid, videod ning kasutaja saab kõiki neid videoid vaadata, laikida ja kommenteerida. Antud rakenduses on vaja teise kasutaja jälgimiseks leida kasutaja profiil, keda soovid jälgida ning vajutada

nuppu *follow*. Seda tehes on võimalik selles vaates näha, kui see inimene on midagi uut postitanud.

- Teises vaates on kõik videod sõltumata sellest kas sa järgid neid inimesi või mitte ning need videod on järjestatud laikide põhjal, kõige rohkem laigitud video kõige ees. Selles vaates saab otsida ka teisi kasutajaid.

- Kolmandas vaates saab kasutaja valida, kas ta lisab foto oma galeriist, teeb uue pildi või lindistab video. Pildi tegemisel või video filmimisel saab ta ka panna videole pealkirja ja selle siis üles laadida.

- Neljandas vaates näeb kasutaja kõike, mis on tema kontoga seoses toimunud, näiteks kes on teda laikinud või jälgima hakanud.

- Viiendas vaates näeb kasutaja oma profiili. Profiili vaates on kasutaja kõik videod, tema üles laetud videote arv ning teda hetkel jälgivad inimesed. Selles vaates on ka kasutaja seaded.

Selle rakenduse koodi kätte saades läks autoril väga kaua aega, et aru saada kuidas see kogu süsteem toimib, kuna kogu rakendusel puudus dokumentatsioon ning kood oli väga halvasti kommenteeritud.

Androidi rakendus on kirjutatud kasutades põhiliselt kaht erinevat tehnoloogiat: Java ja XML ning ka Androidi SDK. Kood sisaldas üle 300 Java klassi, millest enamus olid pikemad kui 300 rida, ning hulgaliselt teisi faile.

Tagarakendus on kirjutatud kasutades JavaScript-i ning AngularJS-i. Andmebaasiks kasutati Neo4j'd ning sellest pärimiseks Neo4j'le omapärast graafi päringuid.

Kahjuks ei saa siia tuua pilte, et näidata milline see rakendus alguses välja nägi, kuna andmebaas on muutunud ja selle rakenduse esialgne versioon ei tööta praeguse andmebaasiga.

2.3 Autori roll rakenduse arenduses

Autor alustas selle tööga 15.04.2018 ja tegi iga kuu vähemalt 160 tundi tööd, antud töö eest sai autor ka palka. Idee algataja ning ka selle projekti ülemus, Arnet Mahmastol, jättis

autorile vabad käed selles osas, mis tehnoloogiaid ta antud tööl kasutada võib ning kuidas ta probleemi lahendab. Autoril oli täielik vastutus rakenduse eest ning Arnet lasi autoril enamus valikud ise teha.

Selles töös pidi autor algselt aru saama, kas on üldse võimalik taolist rakendust teha. Selle ülesande positiivsel tulemusel pidi autor ühendama oma tehtud prooviprojekti ja Picogram-i. Seejärel muutma kogu rakenduse välimust, suure osa funktsionaalsusest ning ka andmebaasi ja tagarakendust. Seejuures pidevalt täiendades rakenduse põhiosa ehk laulmise osa.

Kuna projekt oli edukas, kaasati ka IOS-i arendaja, kes arendas sama rakendust IOS seadmetele, kasutades autori tehtud algoritme ja funktsionaalsusi. Autorist sai ka ülalmainitud IOS-i arendaja projektijuht.

Antud bakalaureuse töös keskendub autor põhiliselt rakenduse kolmandale vaatele ehk laulmise osale, kuna see oli väga suur töö ja kogu projekti üles kirjutamine läheks liiga mahukaks.

2.4 Olemasolevad lahendused ja konkurendid

Sellist tüüpi rakendusi leidub turul väga vähe. Autorit inspireeris laulmise vaate tegemisel mäng nimega Guitar Hero 5 [4], kus kasutaja saab endale osta mikrofoni ning läbi selle laulda nähes sõnu nagu seda on kujutatud Joonis 1, lk 14.

Joonis 1: Guitar Hero 5 vokaali vaade

Antud jooniselt on näha rohelise joonega kui kaugel laul antud hetkel on ning kust kasutaja peab laulma. Tumedam kollane joon näitab kui kõrgel on antud hetkel kasutaja helikõrgus ning tühjad ruudud, mis jäävad rohelisest joonest paremale poole näitavad kui kõrgelt peab kasutaja laulma.

Guitar Hero 5 ja ka teised Guitar Hero versioonid olid väga edukad ja kuulsad mängud, kuid ühtegi neist ei olnud tehtud Androidi seadmele. See tähendab, et meie nendega ei konkureerinud, aga samas andis see rakendus autorile häid ideid.

Otseseks konkurendiks oli rakendus nimega Smule. Smule on suhteliselt suure kasutajate võrgustikuga rakendus, mis on tehtud nii Androidi kui ka IOS platvormile. Rakenduse Smule põhimõte ning vaated ja funktsionaalsus olid täpselt sama nagu rakenduses, mida autor arendas.

Smule-i ja StarMe kõige suurem vahe on selles, millisel viisil rakendused kuvavad kasutajale kuidas ja kust laulda ning millist tagasisidet nad kasutajale annavad. Smule oli tehtud väga lihtsal viisil, kus kõrgused ei olnud väga täpsed ja terved laused olid ette antud ühel kõrgusel. Selline lahendus tagas küll stabiilse vaatepildi, kuid see ei aidanud kasutajal õigesti laulda ning tagasiside, mida kasutaja sai oma laulmisest, oli ka üsnagi kesine. StarMe kõige suurem fookus oli teha perfektselt funktsioneeriv laulmise osa, kus kasutaja näeks täpselt kui kõrgelt ta antud hetkel laulab ning saaks ennast koheselt parandada.

3 Funktsionaalsed nõuded

Nagu on ka eelnevalt mainitud, siis edaspidi keskendutakse antud bakalaureuse töös keskmisele vaatele ehk laulmise osale ja seetõttu nimetab autor ainult keskmise vaatega seotud funktsionaalsed nõuded. Funktsionaalsed nõuded on ka kujutatud tegevusskeemil Joonis 2, lk 18.

1. Vajutades keskmisele ekraanile saab kasutaja valida väga paljude laulude seast endale meeldiva laulu. Laulud on jagatud žanride kaupa erinevatesse kategooriatesse. Enne, kui kasutaja saab laulu laulmisega alustada, peavad olema täidetud järgmised tingimused:

    a. Kasutajal peab olema laul ostetud rakenduse krediidi eest. Rakenduses on krediidisüsteem, mida kutsutakse eesti keeles tähtedeks. Ühe laulu ostmine maksab 2000 tähte. Tähti saab kasutaja teenida kui lindistab ennast laulmas ning kui ta on seda piisavalt hästi teinud ehk suurema osa nootidest õigesti laulnud.

    b. Kui laul on enda kontole ostetud, peab kasutaja laulu alla tõmbama. Laulu alla tõmmates jääb laul kasutaja telefoni nii kauaks kuni kasutaja otsustab selle ära kustutada.

    c. Kui laul on ostetud ja alla tõmmatud, saab kasutaja valida laulu ning vajutada „Lindista“ nuppu.

2. Olles laulu valinud ning „Lindista“ nuppu vajutanud, avaneb kasutajale vaade, kus ta näeb järgmisi detaile:

    a. Laulu sõnad: Kõige üleval ekraani tipus näeb kasutaja laulu sõnu, mis aitavad tal laulu kaasa laulda. Sõnad on jagatud eraldi lauseteks, mida laulu sõnade vaade toob eredama tooniga esile kui laulus hakkab uus lause.

    b. Kaamera: Kasutaja näeb kogu tausta katvat lindistust, mida telefoni esikaamera lindistab.

c. Kõrguste meeter: Ekraani keskel vasakus ääres näeb kasutaja kõrguste meetrit, mis püsib alati paigal ning näitab kasutajale millisel hetkel peab ta laulma, et laulu õigesti laulda.

d. Aeg: Vasakus alumises nurgas näeb kasutaja aega, kui kaua on ta juba antud laulu lindistanud.

e. Mängi-paus nupp: Keskmises alumises ääres näeb kasutaja suurt mängu nuppu, millele vajutades hakkab lindistamine tööle. Kui mängi nupule on vajutatud, ilmub paus nupp. Paus nupule vajutades lõppeb laulu lindistamine.

f. Kasutaja tähed: Paremas alumises nurgas on kasutaja tähed, mis näitavad kasutajale mitu tähte ta on selle laulu jooksul teeninud.

3. Laulmise vaates saab kasutaja vajutada ainult „mängi“ nuppu. Nupule vajutades hakkab mängima laulu taustamuusika, ehk laul ilma vokalisti hääleta, ekraani keskel hakkab jooksma joon, mis näitab kasutajale kust tuleb laulda ning laulu sõnad hakkavad vahetuma, et näidata kasutajale mida laulda.

4. Kui laul on lõppenud või kui kasutaja vajutab stop nupule, mis ilmub mängi nupu asemele kui sellele on korra vajutatud, lõppeb laulu lindistamine ning avaneb uus vaade, mis sisaldab endas järgmist:

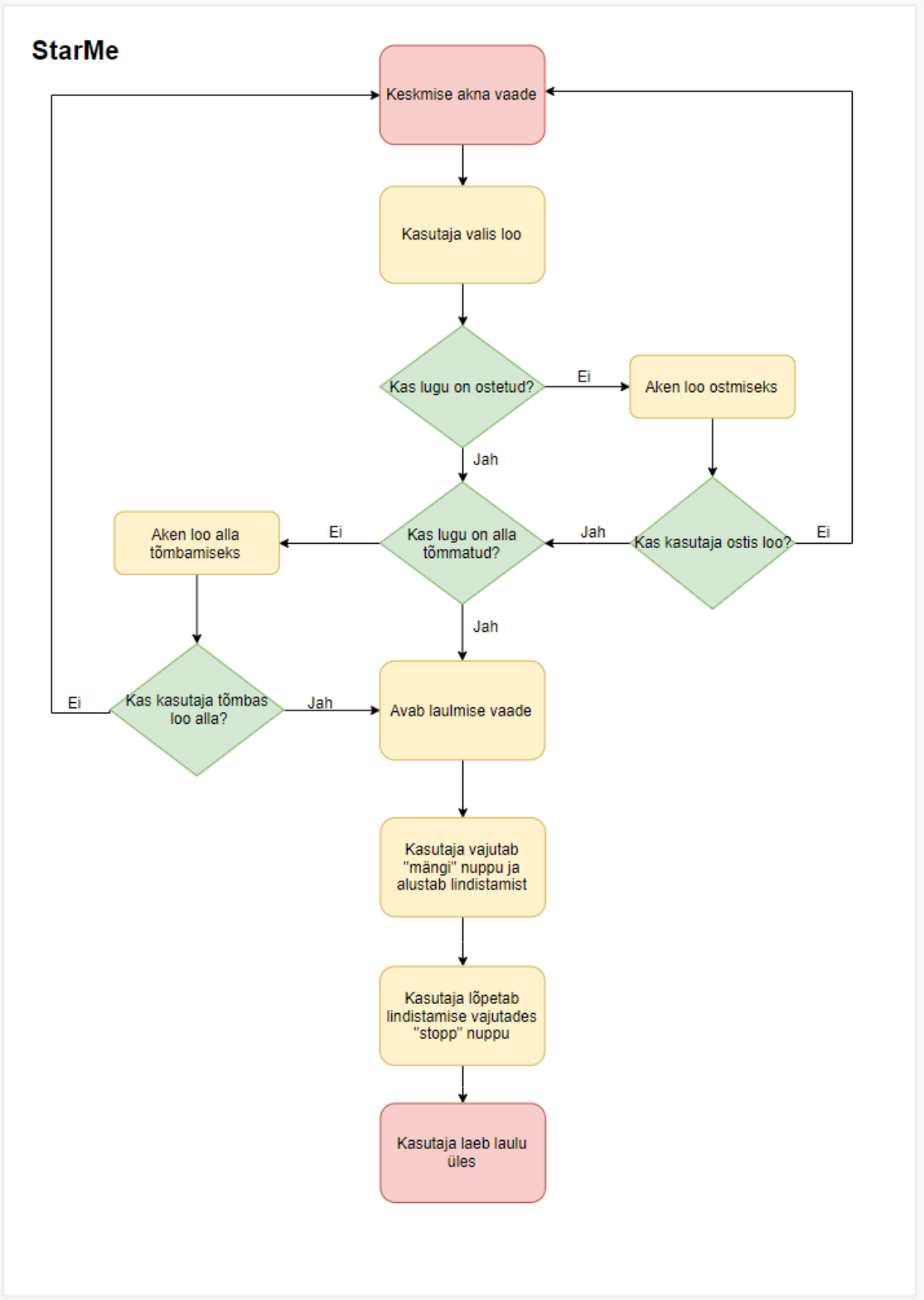
    a. Element kuhu sisestada oma video pealkiri

    b. Video nupp, kuhu peale vajutades näeb ja kuuleb kasutaja mida ta just lindistas ja saab seda edasi-tagasi kerida.

    c. Element, mis näitab kui palju tähti kasutaja teenis selle laulu jooksul. Sellele peale vajutades näeb kasutaja kui täpselt ta laulis (eri värvi tähed näitavad kui täpselt ta laulis teatud noote).

    d. Element, kus kasutaja saab valida oma videole filtri. Filtri peale vajutades muudetakse videot koheselt ning uuesti video nupu peale vajutades näeb kasutaja milline ta video välja näeks koos uue filtriga.

e. Üles laadimise nupp *Release*, mille peale vajutades laetakse video üles andmebaasi ning kõigile teistele kasutajatele nägemiseks.



Joonis 2:Tegevusskeem keskmise vaate lindistamise protsessist

4 Testrakenduse arenduskäik

Selles peatükis kirjeldab autor kuidas rakenduse arendus algas, milliseid tehnoloogiaid ja lahendusi ta kasutas ning kuidas ta lõpuks lahenduseni jõudis.

Kuna autor ei olnud algselt kindel, kas sellise rakenduse arendamine on võimalik ning ei teadnud kuidas seda teha. Selleks pidi ta koostama alguses testrakenduse, mille nimeks sai *singingPitches*.

4.1 Esimene katsetus

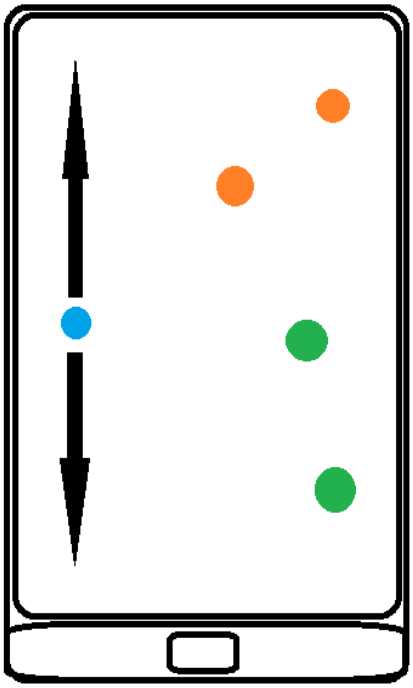
Testrakendus oli tehtud kasutades keelt Java ja XML. Autor kasutas arendamisel Android Stuudiot, mis on platvorm just Androidi rakenduste tegemiseks.

Esimese katsena tegi autor lihtsa rakenduse, kust pallid liikusid paremalt vasakule ning kasutaja tegelane pidi neile pihta saama.

Joonis 3, lk 20 on näha oranže ning rohelisi pallikesi. Nad liiguvad paremalt vasakule ning kasutaja tegelane on sinine pall, millega saab liikuda üles ja alla. Esialgne versioon oli tehtud vajutusega. Ehk kui kasutaja vajutas ekraani peale, liikus pall kõrgemale ja lahti lastes kukkus tagasi allapoole.

Selline lahendus ei sobinud lõpliku soovitud lahendusega ning seda tuli muuta.

Järgmise sammuna pani autor palli liikuma hääletugevuse järgi.



Joonis 3: Testrakenduse esialgne välimus

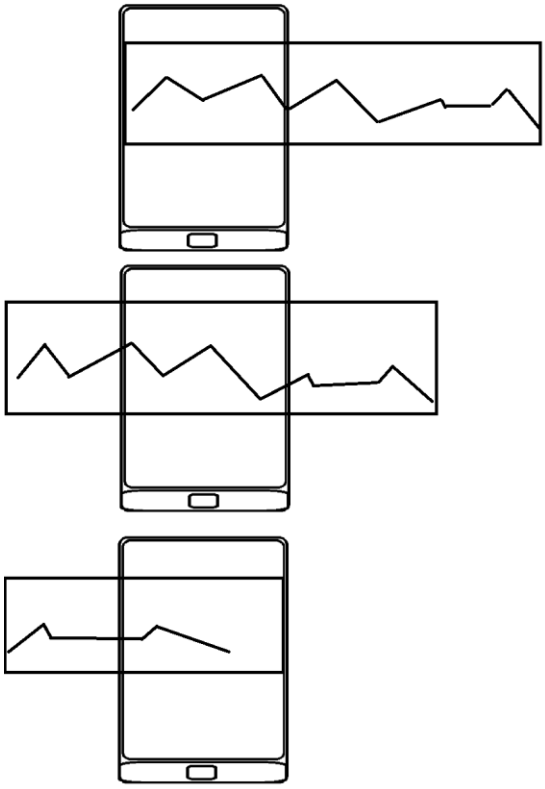
Kui kasutaja tõstis häält siis liikus pall üles ja kui kasutaja ei tekitanud häält, kukkus pall alla tagasi. Selline lahendus ei olnud ka õige lõpliku lahendusega, kuna hääletugevuse järgi ei saa inimest õigesti laulma panna, selleks oli vaja kasutaja vokaali helikõrgust, aga seda kirjeldatakse käesolevas töös hiljem.

Sellise testrakenduse tegemine võib tunduda tühine, kuid andis autorile väga palju uusi ideid ja mõtteid kuidas antud rakendust arendada. Selle testrakenduse arenduse käigus sai autor teada, mida ta täpsemalt tegema pidi ja selle tõttu oli järgmiseks sammuks pallide asendamine kuna nende järgi ei saa laulda.

4.2 Laulu graafiline esitus

Probleemiks oli, kuidas esitada kasutajale pidevalt liikuvat joont, mille järgi ta laulda saaks. Ülesande juures valmistas autorile raskusi see, et joon pidi pidevalt liikuma ühest ekraani äärest teise ja olema koguaeg erinev.

Peale paljude lahenduste proovimist ja otsimist, otsustas autor, et kõige mõistlikum oleks teha graafik, ning seejärel see liikuma panna nagu seda on kujutatud Joonis 4, lk 21.



Joonis 4: Graafiku liikumise kujutis

Joonis 4, lk 21 on näha, et on tehtud pikk graafik, mis oli esialgselt täidetud suvaliste numbritega ning see liikuma pandud. Kasutajale näib nagu joon oleks koguaeg pidev ning see oli juba midagi, mille järgi oleks kasutajal võimalik laulda.

Graafiku tegemiseks kasutas autor Android *GraphView-d* [5], mis lubas *teha LineGraph-i*, milleks oli punktide järgi tehtud graafik. Autor pidi sisestama punkti X ja Y koordinaadid ning nad seejärel ühendama. Ja tuligi pikk graafik nagu seda on kujutatud Joonis 4, lk 21.

Järgmiseks sammuks oli graafiku liigutamine. Seda sai teha vahetades iga teatud ajavahemiku tagant graafiku *ViewPort*-e, mis olid graafiku algus X ja lõpp X koordinaadid, mida vahetades muutus millist osa graafikust hetkel näidatakse. Näiteks prooviks seadis autor algus X koordinaadi nulliks ning lõpp X koordinaadi üheks ja muutis seda iga sekundi tagant ühe võrra. Ehk 1 sekundi möödudes oli algus X koordinaat üks ja lõpp koordinaat kaks. Selline lahendus jõnksutas graafikut väga palju ja graafiku liikumine ei olnud sujuv, aga sobis antud hetkeks. Järgmine samm oli graafik täita reaalsete väärtustega laulust.

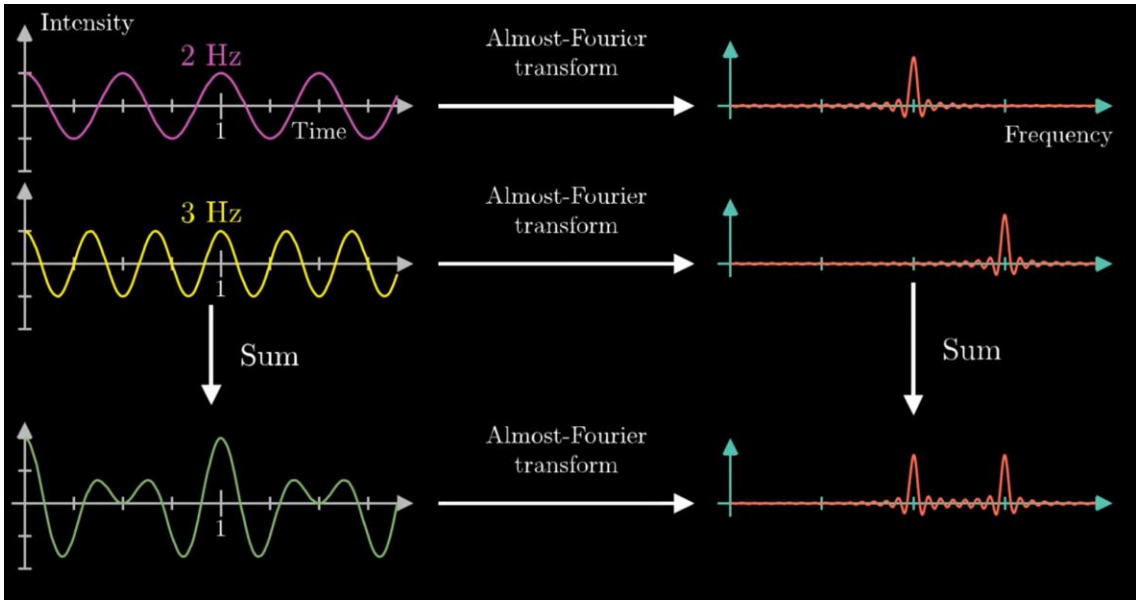
4.3 Graafiku täitmine reaalsete laulu helikõrgustega

Laulust helikõrguste kätte saamine oli selle rakenduse juures üks raskemaid samme. Pika uurimise järel oli autorile selge, et selleks tuleb kasutada FFT valemit. FFT valemist täpsemalt räägib autor peatükis 4.3.1.

Autor veetis terve kuu proovides ise FFT valemit teostada, selle kohta võimalikult palju õppida ning katsetada olemasolevaid lahendusi ehk API-liideseid. API-liides on rakendusprogrammiga määratud reeglistik, mille alusel kasutab operatsioonisüsteem teise rakendusprogrammi teenuseid. Enne jätkamist tuleks FFTst kiire ülevaade teha.

4.3.1 FFT, YIN ja YinFFT

FT ehk Fourier' teisendus on signaalitöötuses üks enim kasutatud mõistetest. Fourier' teisendus võtab antud ajahetkel esinevad helid ja muundab nad sagedusteks. Antud kontekstis saab FT kohta öelda ka DFT ehk diskreetne Fourier' teisendus, kuna tegeletakse ainult diskreetsete arvudega. Hea ülevaate sellest saab Joonis 5, lk 22 [6].

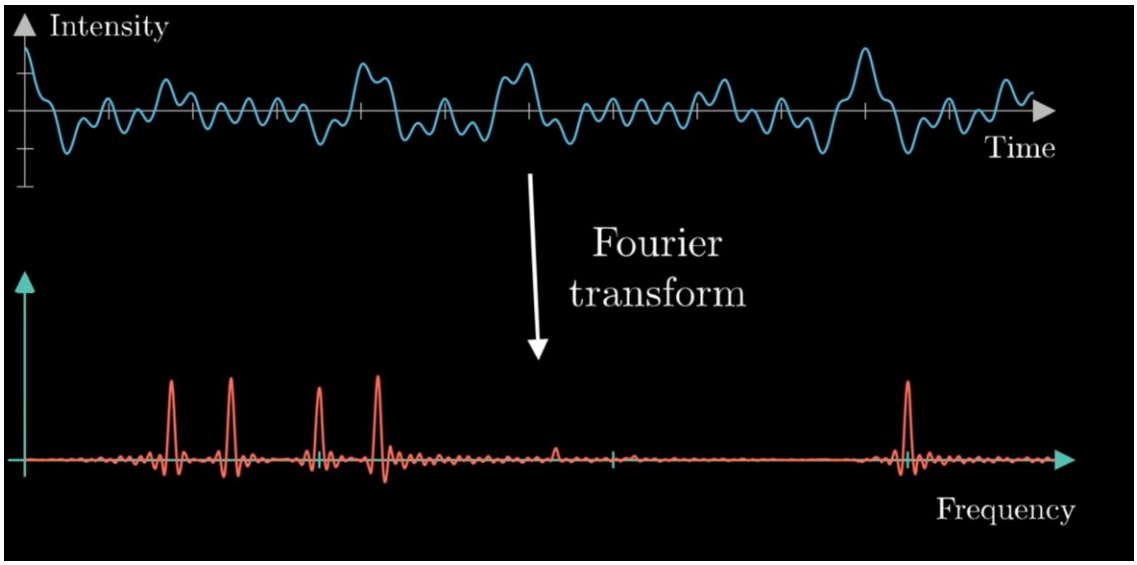


Joonis 5: FT näide

Sellel joonisel vasakul poolel on näha heli ilma, et seda oleks töödeldud kasutades DFT valemit. Alguses antakse sisse (joonisel lillaga) ühe heli siinuslaine ning DFT leiab selle helisageduse hertsides. Samamoodi on näha ka seda kollase joonega, kus võetakse teistsugune helilaine, rakendatakse DFT valemit ja saadakse teine helisagedus. Nüüd kui need kaks omavahel kokku liita üheks heliks (lilla ja kollane), siis DFT leiaks ikkagi

sellest ühest helisignaalist 2 erinevat sagedust. Nagu seda on näha joonise paremas alumises nurgas oleval joonel.

Seda kasutatakse eelkõige helitöötlusel, kus näiteks mingis helifailis on kõrge pinin või mingi muu heli, mida soovitakse eemaldada. Alguses muudetakse helilaine DFT abil sageduse graafikuks ning seejärel eemaldatakse sageduse järgi see heli, mida sooviti. Joonis 6, lk 23 [6] on näha, kuidas muudeti üks helilaine, kasutades DFT valemit, sagedusgraafikuks, mille pealt on lihtne visuaalselt või andmete järgi näha, milline heli on üleliigne. Antud näites on selleks paremal asuv kõrge sagedus ning selleks, et lahti saada segavast sagedusest, tuleb see sellest helifailist eemaldada. FFT ehk kiire Fourier' teisendus on keerukuselt lihtsam ning seetõttu ka kiirem versioon DFT-st, mistõttu leiab see rohkem kasutust.

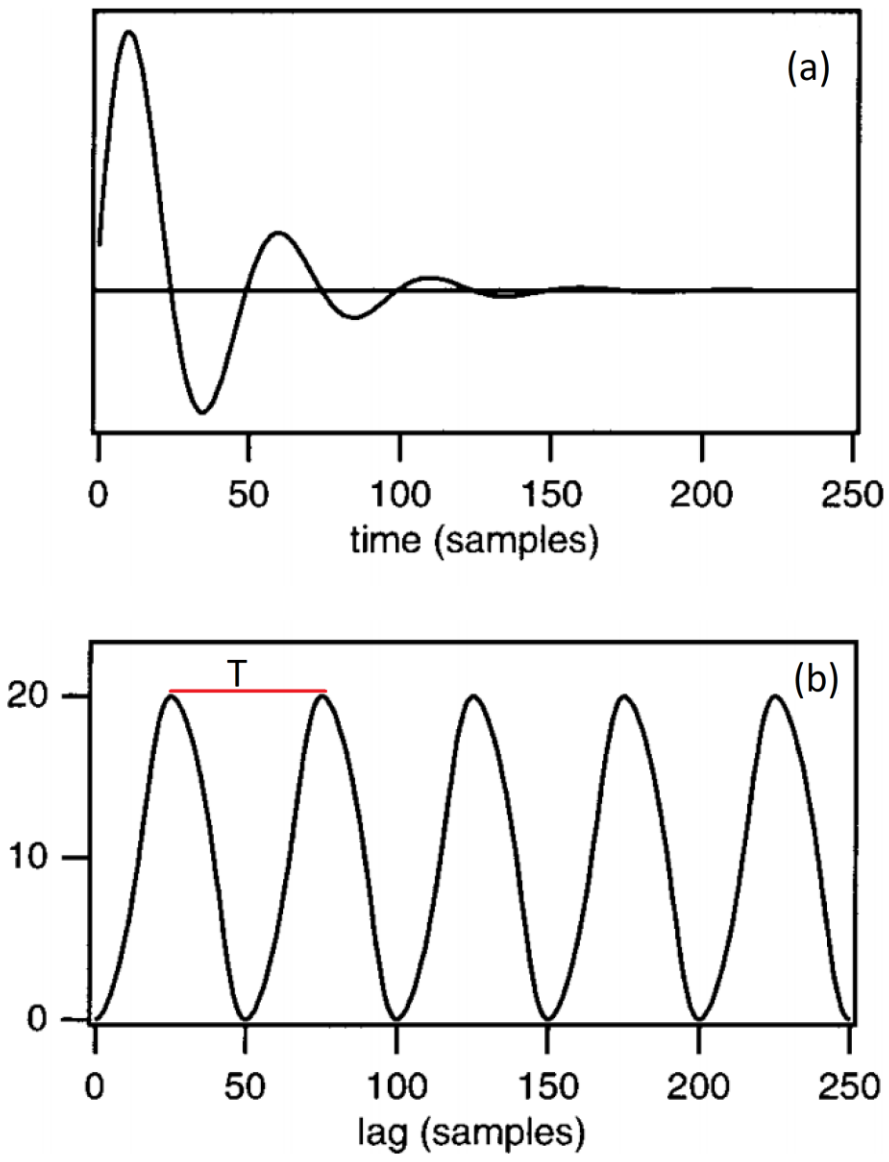


Joonis 6: DFT kõrge sagedus

On mitmeid helikõrguse tuvastamise algoritme [7], kuid ükski neist ei ole töö kirjutamise hetkel piisavalt täpne, et seda saaks nimetada perfektseks, ning et see leiaks rakendust igas valdkonnas. Kuna autor kasutab enda töös TarsosDSP API-liidest [8], ning see omakorda kasutab antud ajahetkel helikõrguse leidmiseks YinFFT helikõrguse tuvastamise algoritmi, siis räägime lühidalt ka YIN helikõrguse tuvastamise algoritmist.

YIN helikõrguse tuvastamise algoritmi [9] kasutatakse eelkõige kõne ja muusika jaoks. YIN meetodit on hea kasutada kuna uuringu tulemusena [9] selgus, et just sellel meetodil on kõige väiksem vigade protsent. Nagu sai ka eelnevalt mainitud, siis tänaseni pole veel leitud ühtegi meetodit, mis oleks 100% täpne. Selle tõttu tuleb kasutada neid, mille vigade

protsent on kõige väiksem. Erinevalt teistest helikõrguse tuvastamise algoritmidest, ei ole YIN meetodil vaja ülemist kõrguse limiiti, ning seetõttu on seda hea kasutada just laulude analüüsimiseks, kuna hääled võivad olla väga erinevad ning kõrguselimiidi seadmisel võiksid mõned tähtsad helid kaduma minna. YINi põhifunktsionaalsus seisneb selles, et ta võtab siinuslaine, rakendab erinevus funktsiooni (*Difference function, periodic model with timevarying amplitude*) ning tulemuseks saab laine, mille pealt on juba lihtne arvutada perioodi ja sealt edasi helikõrgust hertsides. Seda protsessi on kujutatud Joonis 7, lk 25, kus (a) on siinuslaine, millel otsitakse sagedust ning (b) on laine, millel on rakendatud erinevus funktsiooni. Tuleb tähele panna, et see on lihtsustatud seletus YIN meetodist ning tegelikkuses toimub seal vahel ning ka edaspidistes arvutustest rohkem protsesse.



Joonis 7: Yin erinevusfunktsioon

YinFFT [10] on optimiseeritud ning kiirem versioon Yin algoritmist. YinFFT algoritm on tuletatud YINi algoritmist, mis kasutab FFT valemit, et arvutada kitsendatud ruutude erinevus funtksioon (*tapered square difference function*), millest omakorda on perioodi leidmine kiirem ning lihtsam.

4.3.2 Laulust helikõrguste saamine FFT abil

Autor kasutas olemasolevat API-liidest nimega TarsosDSP [8]. Nimetatud API-liidesel oli funktsionaalsus, mille võis anda sisse laulu, mis oli *WAV* formaadis ja sai tagasi listi, mis koosnes millisekunditest ja laulu helikõrgustest. Selle API-liidese abil koostas autor eraldi Java rakenduse nimega AudioToHz.

AudioToHz oli programm, millele oli autor ka teinud graafilise liidese, kuhu sai sisendiks anda laulu faili *WAV* formaadis ning tagasi sai *txt* formaadis faili. Osa sellest, mida sisaldas *txt* formaadis tagasi tulnud fail, on näha Joonis 8, lk 25.

```
3.99:262.39
4.04:261.22
4.09:262.56
4.13:255.11
4.18:234.58
4.23:224.81
4.27:215.15
4.32:217.12
4.37:221.07
4.41:221.61
4.46:220.64
4.50:219.66
4.55:219.40
```

Joonis 8: Osa AudioToHz txt faili väljundist

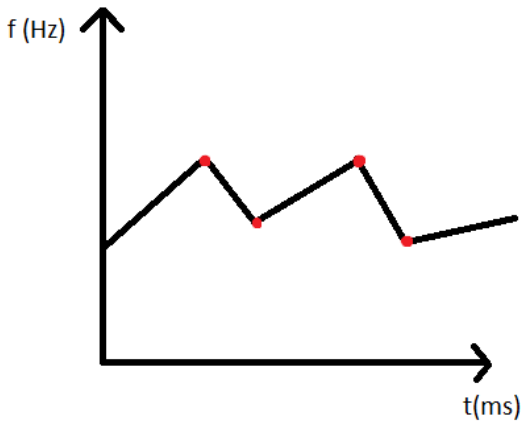
Jooniselt on näha erinevaid ridu ning nendel olevaid numbreid. Seletamiseks kasutame ainult esimest rida, kus on näha järgmist: 3.99:262.39. Esimene pool enne märki „:“ on aeg millisekundites ja teine pool peale märki on helikõrgus hertsides (Hz) ehk antud näites on ajahetkel 3.99 laulu helikõrgus 262.39.

Selleks, et saada täpset tulemust, ning et saadud tulemus ei oleks segatud laulu instrumentaalist või mõnest muust helist tuleb laulu helikõrguste eraldamisel kasutada audio faili, millel on ainult soovitud laulu laulja häält kuulda. Sellised audiofaile saab kui osta endale lugu, millelt soovitakse kõrgusi mõõta.

Ülal nimetatud *txt* failis olid kogu laulu ajal esinenud helikõrgused ja nende ajad ning just selle tulemusega. Järgmiseks sammuks oli saadud tulemused panna graafikusse.

4.3.3 Saadud helikõrguste lisamine graafikusse

Saadud helikõrguste lisamine käis järgmisel põhimõttel. Saadud helikõrguse ajahetk (ms) on punkti X koordinaat ning helikõrgus (Hz) on punkti Y koordinaat. Selline lahendus on kujutatud Joonis 9, lk 26.



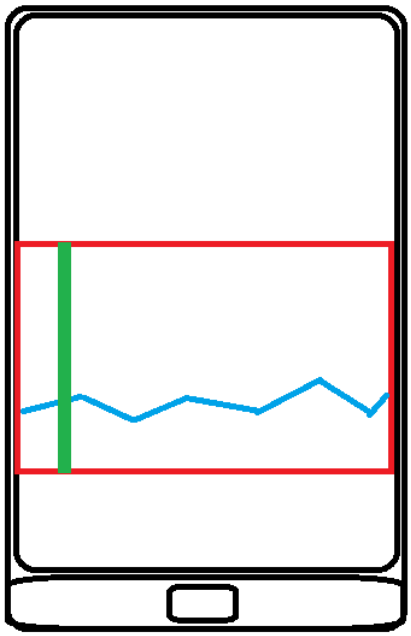
Joonis 9: Sageduste ja aja sõltuvuse graafik

Joonisel on näha punaste täppidega kujutatud erinevatel ajahetkedel esinevaid helikõrgusi ning nendest läbitõmmatud joont. Sellisel moel luges programm sisse kõik sisendid *txt* failist ning koostas sellest graafiku. Kui laulu graafik oli valmis, tuli sisse lugeda kasutaja hääle sisendit helikõrgustena, need ekraanil kuvada ning kontrollida, kas kasutaja laulab õigesti.

4.4 Kasutaja sisendi lugemine

Kuna autor oli kasutanud helikõrguste lugemiseks laulust API-liidest TarsosDSP-d, siis tuli ka kasutaja hääl lugeda kasutades sama API-liidest, et tulemused oleksid võimalikud täpsed.

Selleks, et kasutaja teaks kuna ja kust ta peab antud hetkel laulma, tuli teha ekraanile joon, mis näitaks kasutajale, kus antud hetkel graafik on ja kust ta laulma peab. Sellist esitust on näha Joonis 10, lk 27.



Joonis 10: Kasutaja joon näitamaks, kust tuleb laulda

Joonis 10, lk 27 on näha punasega graafiku ala, kus jookseb sinisega joon. See näitab kus tuleb laulda ja rohelisega on näha kasutaja joont, mis näitab mis ajahetkel peab kasutaja antud kõrgusel laulma.

Selleks, et tagada, et kasutaja hääle sisend oleks võimalikult täpne ja mitte häiritud kõrvalistest teguritest, soovitas rakendus kasutajal laulda kõrvaklappidega, et ei oleks kuulda laulu taustamuusikat ning laulda vaikses toas.

Kasutajale tagasiside andmiseks tema laulmisest tuli kuvada kasutaja nupp ülalmainitud rohelise joone peal, mis liiguks üles-alla vastavalt kasutaja häälekõrgusele. Testrakenduseks oli kasutaja nupuks pallike, mis liikus mööda rohelist joont vastavalt kasutaja häälekõrgusele. Selleks hetkeks oli valmis kasutaja nupu liigutamine joonel ning graafiku liikumine, kuid nad ei olnud sünkroonis ja kasutaja ei saanud tagasisidet, kas ta laulab laulu õigesti. Seetõttu tuli graafik panna liikuma õige kiirusega.

4.4.1 Graafiku liigutamine laulu kiirusel

Graafiku liikumise kiiruse arvutamine oli raske protsess, kuna tuli arvestada, et laulu alguspunkt on kasutaja joone asukoht.

Nagu ka eelnevalt mainitud, sai graafikut liigutada tema *ViewPortid-e* algus X ja lõpp X koordinaate vahetades. Graafik tuli ehitada nii, et graafiku esimene X koordinaat oli

võrdne laulu algusega ja viimane graafiku viimane X koordinaat oli võrdne laulu viimase noodiga.

Testimise tulemusel selgus, et kõige sujuvam graafik tuleb kui kasutajale kuvada korraga 20ms ja seda liigutada iga 20ms tagant. Korraga 20ms kuvamine eeldas, et kui graafiku *ViewPortid-e* X koordinaatideks valida 0 ja 0.2, siis graafik näitab laulust korraga ainult 20ms. Sellise tulemuse saavutamiseks lähtuti sellest, et ekraanil oleks korraga võimalikult väike osa graafikust ning graafikut liigutataks võimalikult tihti. Kuid parim tulemus saavutati ikkagi katse-eksitus meetodit kasutades.

4.4.2 Kasutaja lauldud helikõrguse kontrollimine

Nüüdseks oli juba valmis sujuv graafiku liikumine, mis kuvas kasutajale kust laulda. Valmis oli ka kasutaja häälekõrgusele reageeriv nupp, mis näitas kasutajale kui täpselt ta laulab, aga kasutaja ei saanud mingit tagasisidet oma laulmisest. Niisiis tuli kontrollida kasutaja häälekõrgust graafikul ehk laulus esinevate helikõrgustega.

Antud testrakenduses selleni ei jõutud, ehk seda protsessi kirjeldab autor peale seda kui testrakendus on juba põhirakendusse integreeritud. Testrakenduse korrektsust testis autor oma häälega ja muusikateadmistega.

4.5 Laulusõnade kuvamine

Testrakendusse sai ka lisatud laulusõnade kuvamine. Selle jaoks tegi autor eraldi vaate, mis kuvas sõnu erinevatel ridadel. Hetkel laulus kostvad sõnad kuvati teise värviga kui tulevased read. Kui rida oli ära mängitud, kadus see rida vaatest ning selle asemele tuli uus rida sõnu. Laulusõnade vaade on kujutatud Joonis 11, lk 29.

Laulusõnade teostuseks kasutas autor valmis API-liidest nimega LyricView [11] ning muutis seda vastavalt oma vajadusele. Laulusõnade näitamiseks oli vaja sisestada laulu *lrc* laiendiga fail.

Joonis 11: Laulusõnade kuvamine

4.6 Video kuvamine

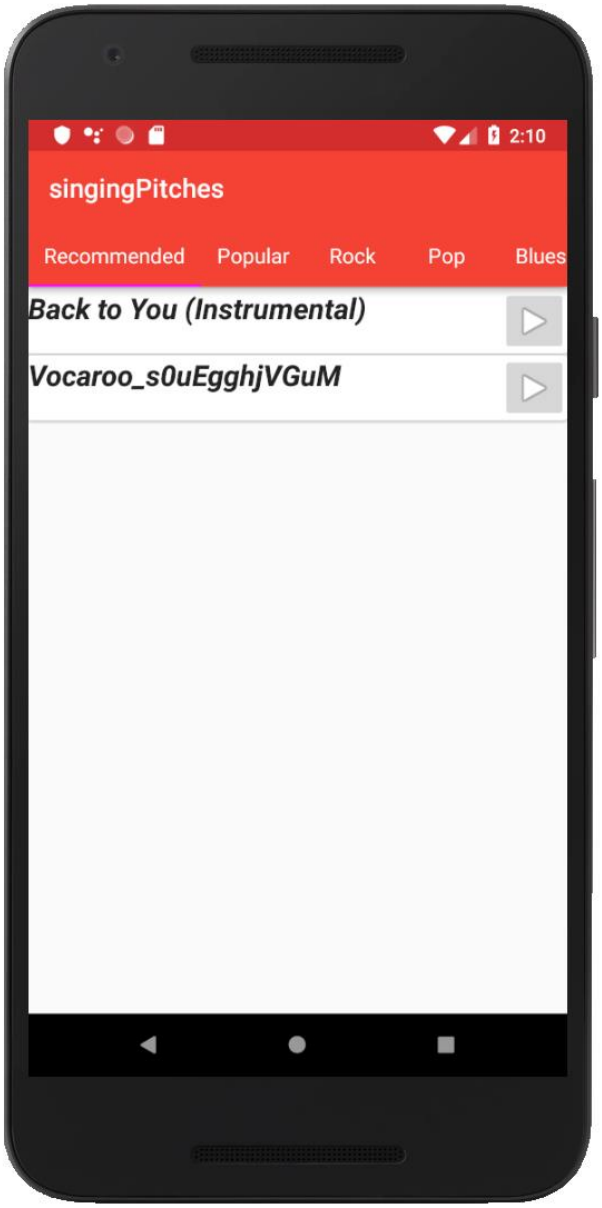
Viimasena oli vaja lisada kasutaja video kuvamine ja selle lindistamine. Video kuvatakse üle terve ekraani teiste vaadete all. Videot lindistatakse ainult esikaamera kaudu, et näeks kasutaja nägu kui ta laulab ja video lindistamine on kohustuslik.

Videovaate teostuse tegi testrakenduse jaoks kasutaja autor *Google*i poolt tehtud Camera2Video API-liidest [12]. Antud testrakendus ei lindistanud ega salvestanud kasutaja videot ega heli vaid oli tehtud ainult testimiseks.

4.7 Testrakenduse lõpptulemus

Kui kõik ülaltoodud protsessid olid läbitud, oli autor kindel, et antud rakendust on võimalik teostada. Peale testrakenduse valmimist ja selle testimist, hakkas autor lisama oma testrakendust Picogram-i rakendusse ja kogu seda protsessi kirjeldatakse peatükis 5.

Pildid valminud testrakendusest: Joonis 12,lk 30 ja Joonis 13.lk 31



Joonis 12: Testrakenduse laulu valiku vaade

Joonis 13: Testrakenduse laulmise vaade

Joonis 13, lk 31 on näha ka ülalmainitud graafikut, joont mille järgi laulda ning graafiku 0 punkti, mis näitab kasutajale kust tuleb laulda ning tagataustal on näha laulja pilti esikaamerast. Ruuduline nupp all keskel on *mängi* nupp, mille vajutamisel hakkab laul käima ja graaf ning laulusõnad liikuma.

5 StarMe rakenduse arenduskäik

StarMe rakendust praegusel hetkel edasi ei arendata, kuid ta on saadaval Google Play poes nimega StarMe. Rakenduse arenduskäik peatati 2019 märtsikuus kuna nii Androidi kui ka IOS versioonid olid piisavalt heas seisus, et neid saaks turustada ja leida neile kasutajaid. Kui leitakse, et rakenduse vastu on huvi ja inimesed hakkavad seda rakendust kasutama, siis alustatakse uuesti rakenduse uuendamise ning arendusega.

Selleks aga, et rakendus saada nii kaugele nagu see praegu on, tuli autoril näha väga palju vaeva. Selles peatükis räägib autor kuidas nägi tema jaoks välja StarMe rakenduse keskmise vaate arendus ja kuidas rakendus jõudis sellisesse faasi nagu see praegu on.

5.1 Testrakenduse lisamine StarMe rakendusse

Enne testrakenduse lisamist Picogram-i lähtekoodi tuli autoril alguses koodiga tutvuda. Peale tutvumist eemaldas autor kogu eelneva funktsionaalsuse, mis oli seotud kolmanda vaatega. Seejärel lisas autor oma koodi rakendusse ning sidus testrakenduse käivitamise kolmanda vaate avamisega. Nüüd Picogram-i käivitades avanes ülaltoodud testrakendus ning nüüd ei saa enam rääkida, et tegu on Picogram-iga vaid nüüdsest oli juba uue funktsionaalsusega StarMe rakendus valmimas.

Järgmisteks sammudeks oli kolmanda vaate funktsionaalsuste täiendamine. Juurde tuli teha veel mitmeid asju: kasutaja laulmise kontrollimine, kogu video ja kasutaja hääle salvestamine ning üheks failiks tegemine, laulude lisamine ja nende kättesaamine andmebaasist, valmis tulemuse lisamine andmebaasi ning rakenduse kasutajaliidese täiustamine.

5.2 Kasutaja laulmise kontrollimine ning tagasiside andmine

Selleks, et kasutaja saaks tagasisidet, kas ta laulab antud laulu õigesti tuli kuidagi kasutajale teada anda, kas ta laulab õigesti või mitte. Esialgselt oli plaanis lisada ainult punktisüsteem, ehk kui kasutaja laulab noodi õigesti, saab ta selle eest punkte ning ta näeb ka enda punkte suurenemas.

5.2.1 Helikõrguste kontrollimine algse laulu helikõrgustega

Kasutaja häälenupp, mis liikus üles ja alla vastavalt kasutaja helikõrgusele töötas õigesti ning ka graafik, mis liikus oli täpne, kuid nende kahe klappimise kontroll osutus raskeks ülesandeks.

Visuaalselt ei saanud neid kontrollida, kuna nad töötasid erinevatel tasanditel ja värvidega põrke tulemused ei olnud väga täpsed, kuna erinevatel telefonidel olid eri paksusega jooned ning kasutaja nupu suurused.

Ainsaks lahenduseks jäi laulu listist õige kõrguse leidmine antud ajahetkel. Ehk selleks, et leida õiget kõrgust kus kasutaja laulma pidi, võeti aeg, kaua laul kestnud oli ning otsiti binaarse otsingu algoritmiga kogu listist antud ajale lähim aeg ja võeti sellel ajahetkel algsest laulus esinev helikõrgus. Näiteks kui laul oli kestnud 4.321ms, otsiti algse laulu listist lähim ajahetk sellele ehk Joonis 8, lk 25 oleks selleks 4.46ms ning sellele vastav helikõrguse väärtus 220.64Hz. Seejärel kontrolliti saadud helikõrgust kasutaja lauldud helikõrgusega. Kui kasutaja lauluhääle helikõrgus oli kas kuni 50Hz suurem või kuni 50Hz väiksem, siis loeti see noot õigeks ning kui oli suurem kui eeltoodud vahe, siis loeti see noot valeks.

Õige noodi laulmise korral sai kasutaja endale kirja 1 punkti. StarMe rakenduses loetakse punkte tähtede süsteemis, ehk 1 punkt võrdub 1 tähega. Sellega oli saavutatud laulmise kontrollimine. Selleks, et anda kasutajale rohkem informatsiooni tema korrektsusest, muutis tema nupp värvi, liikudes samaaegselt joonel üles-alla ning talle tuli ka juurde 1 täht.

5.2.2 Helikõrguse korrektsuse informatiivsemaks muutmine

Väga pikalt piirduti rakenduses ainult sellega, et kui kasutaja lauldud noot oli 50Hz raadiuses algsest helikõrgusest, siis loeti see õigeks, kuid hiljem tehti juurde lisa funktsionaalsus. Uus funktsionaalsus andis kasutajale palju täpsema tagasiside. Täpsem tagasiside tähendas kasutajale, et ta näeb kui tema nupp muudab värvi vastavalt tema helikõrguse täpsusele. Värvid jaotusid järgmiselt:

- Kui helikõrgus on +/- 13Hz, siis näitab halli tähte ja kasutaja saab 4 punkti juurde ning seda kutsutakse rakenduses *Perfect* noodiks

- Kui helikõrgus on +/- 26Hz, siis näitab rohelist tähte ja kasutaja saab 3 punkti juurde ning seda kutsutakse rakenduses *Excellent* noodiks
- Kui helikõrgus on +/- 39Hz, siis näitab oranži tähte ja kasutaja saab 2 punkti juurde ning seda kutsutakse rakenduses *Almost* noodiks
- Kui helikõrgus on +/- 52Hz, siis näitab kollast tähte ja kasutaja saab 1 punkti juurde ning seda kutsutakse rakenduses *Lousy* noodiks
- Kui helikõrguse vahe on suurem kui 52, siis näitab punast tähte ja kasutaja saab 0 punkti juurde ning seda kutsutakse rakenduses *Missed* noodiks

Joonis 14: Täpsed tähtede väärtused

Joonis 14, lk 34 näitab tulemust, mida näeb kui kasutaja on lõpetanud laulu laulmise ning soovib oma punkte vaadata. Antud juhul laulis autor 262 *Perfect* nooti ja 154 *Missed* nooti.

Peale nootide näeb ka veel *Timing* noote. Neid saab juurde kui kasutaja laulab sõna esimese noodi õigeaegselt.

5.3 Helikõrguste jagamine sõnadeks ning nende esitamine graafikul

Testrakenduses jooksis graafiku peal pidev joon ning selle järgi oli kasutajal keeruline laulda, kuna kasutaja ei teadnud millal algab uus sõna ning millal lõppeb sõna. Selleks tuli töödelda helikõrgustest saadud andmeid ning välja sorteerida nendest sõnad. Selleks, et oleks parem seletada, lisab autor ühe 4 sekundilise lõigu laulust *Ed Sheeran – Perfect* vt Joonis 15, lk 35.

```
72  7.52:217.18
73  7.57:46512.24
74  7.62:211.94
75  7.66:223.83
76  7.85:263.93
77  7.89:259.34
78  7.94:263.74
79  7.99:262.47
80  8.03:260.95
81  8.08:259.44
82  8.13:263.93
83  8.17:263.80
84  8.22:262.28
85  8.27:260.80
86  8.31:262.62
87  8.36:262.58
88  8.41:262.14
89  8.45:260.86
90  8.50:262.17
91  8.54:260.82
92  8.68:229.61
93  9.10:46468.02
94  9.15:46515.72
95  9.20:46535.80
96  9.24:234.57
97  9.47:46514.27
98  9.75:46475.77
99  9.85:46475.38
100 9.89:46497.35
101 10.17:46510.02
102 10.26:46505.12
103 10.40:46519.37
104 10.50:46461.36
105 10.54:46463.18
106 10.59:46493.81
107 10.63:46463.38
108 10.68:46507.21
109 10.87:117.55
110 11.05:236.28
111 11.10:233.27
112 11.15:235.62
113 11.19:232.05
114 11.24:233.18
115 11.33:265.58
116 11.42:266.88
117 11.47:275.53
118 11.52:270.37
```

Joonis 15: Helikõrguste pikem näide

Sellel joonisel on näha rea numbrid vasakul ääres halliga. Real 73 on näha helikõrgust, mille väärtuseks on 46512.24, mis on helikõrguse jaoks liiga suur. TarsosDSP [8] andis

tühjad või vaiksed noodid kõrgete helikõrgustena ehk siis autori jaoks tähendas see seda, et sellel kohal ei olnud nooti ning need tuleks välja sorteerida, et joon ei hakkaks liialt palju hüppama.

Autor töötas välja järgmise algoritmi, mis käis üle kogu listi ning kui kohtas helikõrgust, mis oli suurem kui 10000, siis see eemaldati nimekirjast ning seda ei kasutatud ja suurendati tühjade nootide loendurit 1 võrra. Kui vahepeale tuli noot, mille väärtus oli alla 10000, siis tühjade nootide loenduri väärtus muudeti tagasi nulliks. Kui tühjade nootide loendur jõudis 5 peale ehk tuli 5 tühja nooti järjest, siis see tähendas, et sealt katkeb laulus sõna. Kõik eelmised väärtused, mis olid loetud, mis ei olnud suuremad kui 10000, pandi ühe sõnana laulu listi. Siis algoritm otsis järgmise sõna alguse, ehk kui see leidis noodi, mille väärtus oli alla 10000 ning sellest järgmine noot ei olnud üle 10000. Järgmise sõna kontroll oli vajalik sellepärast, et helikõrguste lugemise algoritm tegi vahepeal vigu ning oli vaja ette kontrollida ning vaadata kas tulemus on pidevalt sama. Kui leiti uue sõna algus siis korrati protsessi kuni leiti uus viiene või suurem tühjade nootide hulk. Sellest alates katkestati sõna, lisati eelmised noodid üheks sõnaks ning jätkati kuni terve helikõrguste fail oli läbi käidud. Tulemusena saadi üks list sõnadest, mis algasid teatud ajahetkel ning lõppesid teatud ajahetkel. Sõnade list koosnes omakorda sõnadest, mis koosnesid helikõrguste paaridest (aeg, helikõrgus).

Lisaks tühjade nootide kontrollile kontrollis see algoritm, et järgmise sõna ja hetkel uuritava sõna vahe ei oleks üle 75Hz, kuna see tekitas liiga ebastabiilse graafi, mida oleks raske kasutajal jälgida.

Kui sõnade list oli valmis, siis anti graafikule ette sõnade list, mille alusel tekitati igast sõnast erinev joon. Selle tulemusena oli valmis graafik, kus oli visuaalselt näha vahesid ehk oli näha kust algab ning kust lõppeb laulus esinev sõna. Selline lähenemine lihtsustas lauljal laulusõnade järgi kontrollida millist sõna ta peab sellel hetkel laulma ning kasutajale oli arusaadavam kus algab ja kus lõppeb uus sõna laulus.

5.4 Kogu audio ja video salvestamine üheks failiks

Eelnevalt on autor rääkinud, et kui kasutaja laulab, siis käib taustal ainult laulu taustamuusika ehk instrumentaal, algse laulu laulja häält ei ole kuulda. Lisaks tuleb ka

see heli, mida kasutaja laulab ja viimaseks on video fail, mida esikaamera lindistab. Kõik need 3 oli vaja kokku saada üheks *mp4* failiks.

Üks oluline asi, mis tuleks mainida, on see, et telefonil on ainult üks mikrofon. Antud rakendusel oli vaja kuulata kasutaja heli, et liigutada tema nuppu ning mõõta sellest helikõrgust. Rakenduses oli kaamera, mis lindistas ainult videot ning ei lindistanud heli. Selle tulemusena tuli kaks erinevat faili: video *mp4* formaadis ja heli *pcm* formaadis.

5.4.1 Helifailide kokku ühendamine

Kasutajale mängiti laulmise taustaks helifaili, mis oli *mp3* formaadis, kasutaja heli lindistati ja tuli tagasi *pcm* formaadis. Need kaks heli oli omavahel vaja kokku ühendada, kuid see ei käinud nii lihtsasti kui autor oli arvanud.

PCM on heli tooraine, mis tähendab, et seal on väga palju erinevaid töötlemata andmeid. Sel hetkel kui autor seda probleemi lahendas, ei olnud korralikke pcm konvertereid, mis suudaksid sellest failist mp3 faili teha. Lahendused olid ainult *C++* keeles ning autor ei soovinud seda oma rakendusse kasutusele võtta. Parim lahendus, mis autor leidis, oli muundada *pcm* fail *wav* failiks ning seda edasi töödelda. PCM faili *WAV* failiks muundamisel kasutas autor FFmpeg-Android API-liidest [13]. Sellele liidesele tuli sisendiks anda *pcm* fail ning väljundiks tuli *WAV* fail. Kahjuks ei võimaldanud see liides *PCM* faili muundumist *mp3* failiks.

Lõpptulemusena tuli saada *aac* helifail, et see video *mp4* failiga ühendada. Kõige lihtsam oli *aac* faili teha just *wav* failist ning seetõttu tuli ka taustaheli *mp3* fail muundada *wav* failiks. Õnneks sai ka siin kasutada sama API-liidest FFmpeg-Android. Parameetrina oli *mp3* failiga kaasa antud ka salvestise kogupikkus, et lõigata *mp3* failist see osa ära, mida ei olnud vaja kuulda. Liidest kasutades sai ka kasutaja lauldud helifaili, mis oli juba *WAV*-iks muundatud ja taustaheli muundatud *wav* faili kokku ühendada. Nüüd oli üks *wav* helifail, mis sisaldas nii taustaheli kui ka kasutaja lauldud heli. Viimaseks tuli sama liidest kasutades muundada lõpptulemusena saadud *wav* fail *aac* failiks. Sellega oli audio kokku ühendatud ja muundatud *aac* failiks.

Kogu audio muundamise protsess võtab rakenduses üsnagi palju aega ning ei olnud antud hetkel kõige optimaalsem lahendus. Protsessi mingil määral kiirendamiseks leidis autor lahenduse kuidas kõik ülalmainitud protsessid liita kokku üheks protsessiks kasutades

FFmpeg-Android API-liidest. Selline lähenemine kiirendas helifailide muundamist mõne sekundi võrra, mis võib küll tunduda väikse ajavõiduna, kuid rakendust kasutades on see üsnagi märgatav.

5.4.2 Helifaili ja videofaili ühendamine

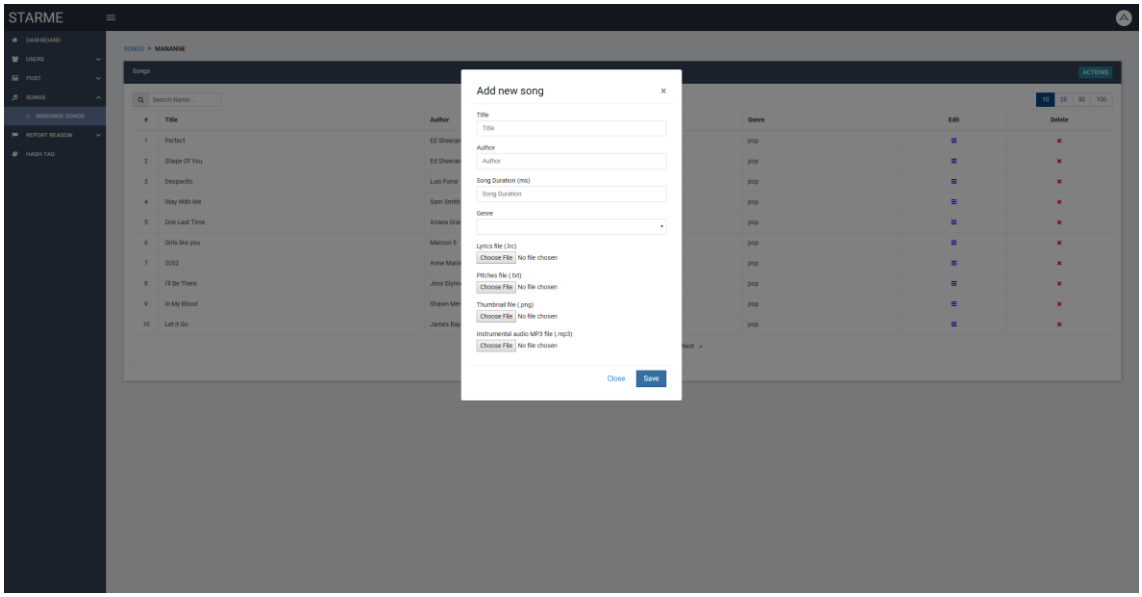
Alles oli jäänud helifaili *aac* ja videofaili *mp4* ühendamine. Selleks, kasutas autor Java MP4 Parser [14] API-liidest. Liidesele anti parameetrina kaasa *aac* ja *mp4* fail ning liides oskas need ise üheks *mp4* failiks kokku ühendada.

5.5 Laulude andmebaasi lisamine

Selle hetkeni oli autor kasutanud lokaalselt juurde lisatud faile, kuid mitut lugu kasutades ei ole see aktsepteeritav lahendus ehk tuli kõik lood ja nendega seotud failid panna andmebaasi.

Looga seotud failideks on: *mp3* taustaheli jaoks, *lrc* laulusõnade näitamise jaoks, *png* laulu pisipildi näitamise jaoks ning viimasena *txt* fail mis sisaldab andmeid algse laulu kõrgustest ning nende ajastustest.

Server asub Ubuntu operatsioonisüsteemis ning autor kasutas failidele ligipääsemiseks rakendust nimega Filezilla [15]. Filezilla ühendas Ubuntu serveriga kasutades *SSH* protokolli. Seal asusid tagarakenduse *JavaScript* failid, millest enamus olid kirjutatud kasutades AngularJS-i. Tagarakendusel oli ka oma veebilehekülg, kust näeb kõikide kasutajate ja nende postituste arvu. Sellele leheküljele lisas autor juurde funktsionaalsuse, mis lubas serveri haldajal sinna lisada laule. Tagarakenduses laulu lisamine nägi välja nagu seda on Joonis 16, lk 39 kujutatud.



Joonis 16: Tagarakenduse laulu lisamise vaade

Laulu lisades tuli sisestada kõik eelnevalt mainitud looga seotud andmed ning laulu pealkiri, autor, laulu kestvus ning laulu žanr. Laul laeti üles Ubuntu serverisse ning andmed nagu pealkiri, autor, žanr läksid ka andmebaasi. Andmebaas on tehtud kasutades Neo4j-d [16], mis on graafidel põhinev andmebaas.

Selleks, et lugusid andmebaasist kätte saada, kasutas Androidi rakendus OkHttp [17] ühendust, ning tagarakenduse Ubuntu serveris asuvaid funktsioone.

5.6 Rakenduse helikõrguse õigsuses veendumine

Kuna rakendus sooviti saada võimalikult täpseks, tuli ka selle rakenduse õigsuses veenduda. Autor on ise tegelenud muusikaga valdava osa tema elust ning kogu rakenduse arenduse käigus kontrollis ta rakenduse õigsuses enda hääle ning muusikateadmistega.

Ühe inimese hääle järgi ei saa veenduda, et rakendus on täpne ning seetõttu tuli ka kasutada teisi isikuid, kes rakendust testiksid. Esimeses rakenduse arendus faasis kasutas autor testisikuteks tuttavaid, kes olid varem laulmisega kokku puutunud. Testiti nii nais- kui ka meeshäältega, et veenduda, et rakendus töötab samamoodi iga häälega. Viimasena veendus rakenduse õigsuses idee autor Tom Veasey, kes nagu ka enne mainitud töötab lauluõpetajana ning hääle treenerina. Tom Veasey testis ka rakendust oma õpilastega, kelleks olid valdavalt lapsed.

Kõikide testgruppide tulemused olid väga positiivsed, ning helikõrgusi mõõtis see rakendus väga korrektselt.

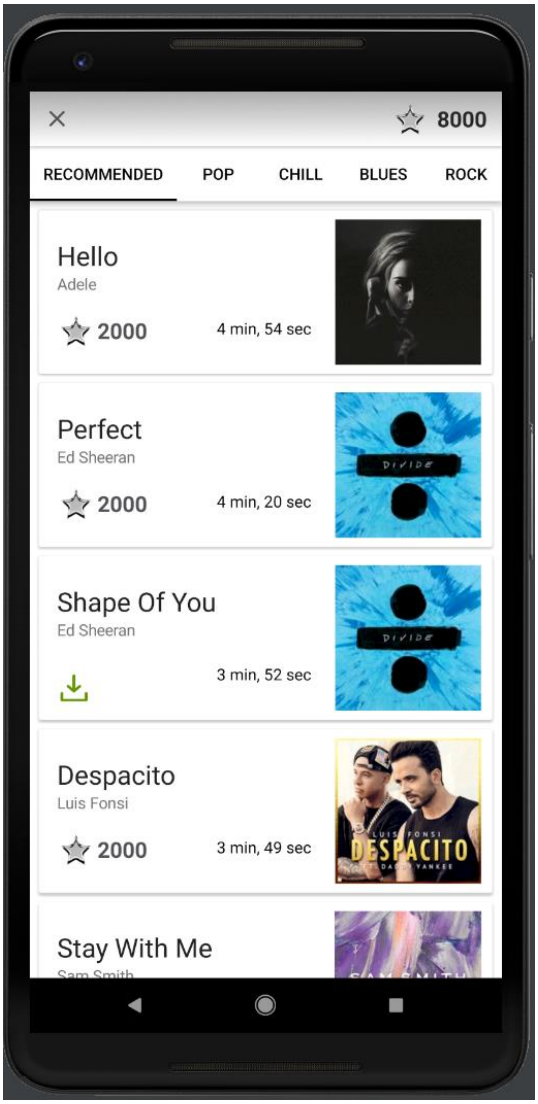
5.7 Rakenduse kasutajaliidese uuendamine

Testrakenduse kasutajaliides oli tehtud ainult funktsionaalsuse näitamiseks ning sellele ei olnud üldse rõhku pandud. Viimase sammuna tuli autoril kogu rakenduse kasutajaliides ümber teha, sealhulgas ka kolmanda vaate kasutajaliides. Kasutajaliidese välimus ei ole selle bakalaureuse töö kirjutamise hetkel ka veel lõplik ning kindlasti muutub kui rakenduse arendus jätkub.

5.7.1 Laulu valiku vaade

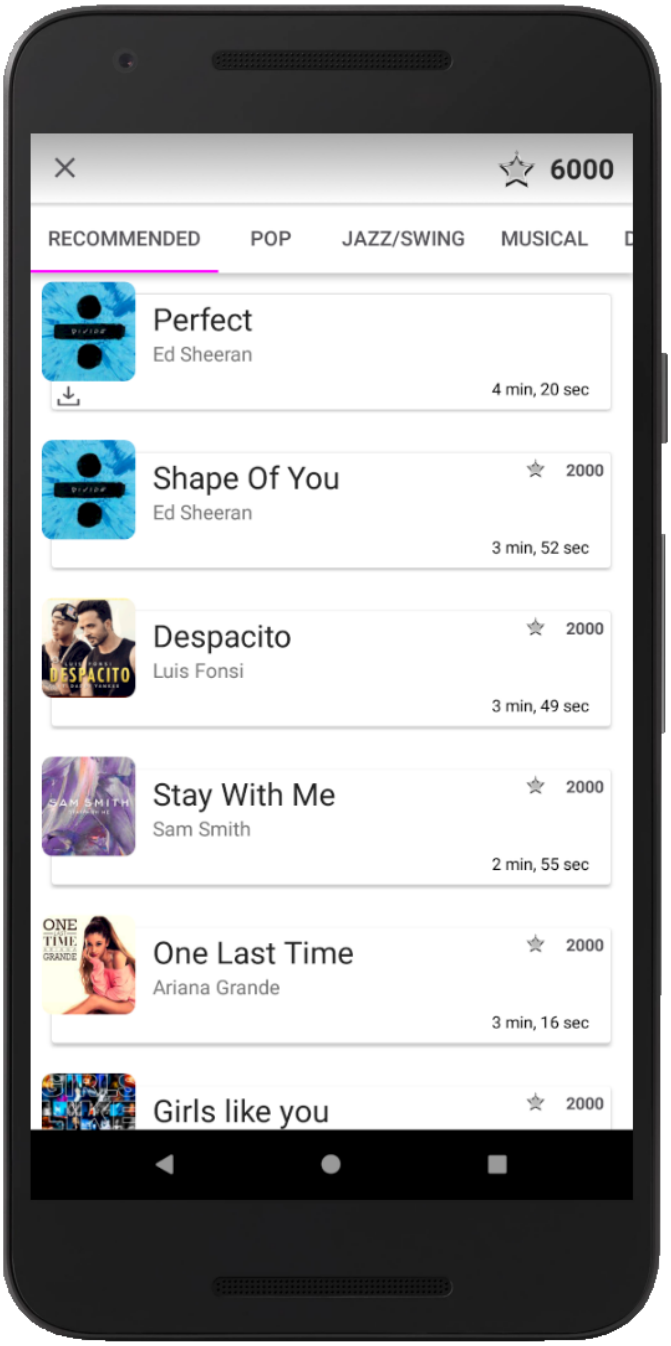
Laulu valiku vaatesse tuli lisada mitmeid uusi funktsionaalsusi, mida ei olnud testrakenduses. Esiteks pidi see näitama kasutaja kogutähtede arvu, teiseks näitama laulu albumi pisipilti, laulu kestvust, laulu hinda ning näitama ka, kas laul on telefoni alla tõmmatud.

Laulu valiku vaadet tegi autor kaks korda ümber. Esialgne vaade nägi välja nagu seda on näha Joonis 17, lk 41.



Joonis 17:Esialgne laulu valiku vaade

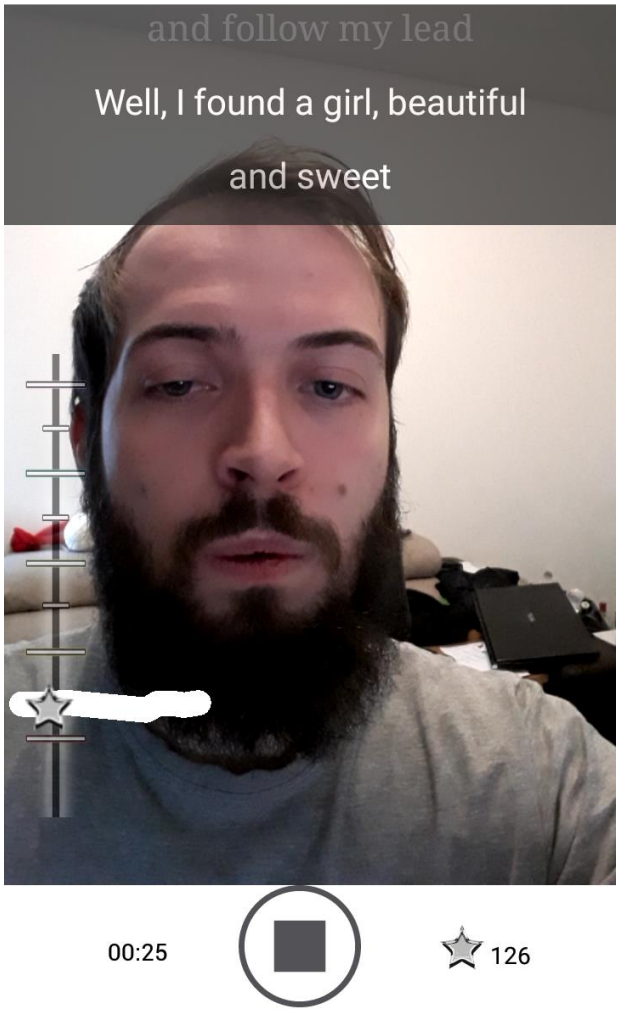
Kuid see ei paistnud just kõige ilusam välja ning see tuli autoril uuesti ümber muuta ning töö kirjutamise hetkeni näeb laulu valimise vaade välja nagu seda on näha Joonis 18, lk 42



Joonis 18:Lõplik laulu vaate valik

5.7.2 Laulmise vaade

Laulmise vaates tegi autor ka omajagu muudatusi. Laulusõnade muutus natukene, graafi ruudustik kadus täiesti ära ning nüüd oli kõik läbipaistev, oli näha ainult posti, mis näitas kuna tuleb laulda. *Mängi-paus* nupp sai uue välimuse ning lisandusid aeg ja kogutud punktide näitaja. Tulemust on näha Joonis 19, lk 43.



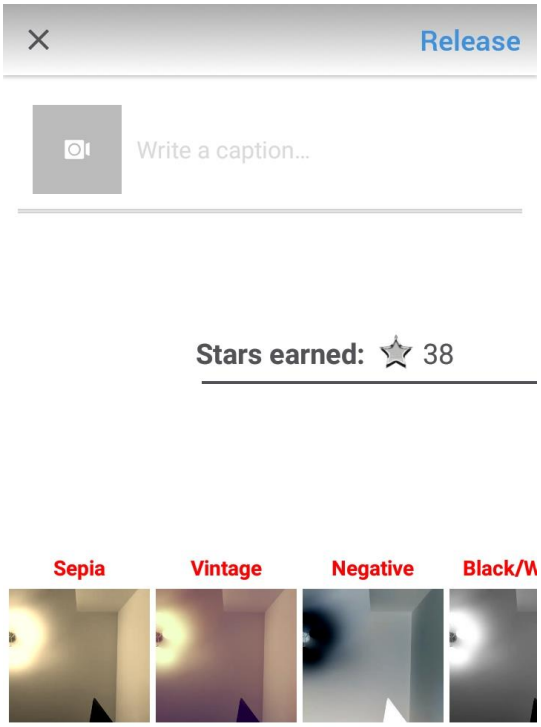
Joonis 19:Laulmise vaade

5.7.3 Laulu salvestamise vaade

Tuli lisada ka uus vaade, mis tuleb peale laulu lindistamist. Ehk vaade, mis tuleb kui kasutaja laulab laulu lõpuni või vajutab *stop* nupu peale. Selles vaates saab lünga „Write a caption“ peale vajutades kirjutada oma postitusele pealkirja. Vajutades kaamerakasti peale avaneb vaade, kust näeb kogu salvestatud videot koos audioga. Vajutades oma teenitud punktide peale avaneb vaade, kust näeb kui palju noote pihta lauldi ja kui täpsed need olid. Viimasena saab valida enda videole filtri.

Filtrid on selles rakenduses tehtud kasutades eelnevalt mainitud FFmpeg-Android-i ning nad muudavad kogu video värvid valitud filtri värviks.

Laulu salvestamise vaadet on näha Joonis 20, lk 44.



Joonis 20: Laulu salvestamise vaade

Sellega piirdus Androidi keskmise vaate arendus. Lisa 1 all on toodud pildid igast rakenduse vaatest ning nad lahti seletatud.

6 Kokkuvõte

Eesmärgiks oli realiseerida Androidi rakendus, mis oleks koduplatvormiks uutele lauljatele ning aitaks uutel lauljatel laulmist õppida. Enne rakenduse teostamist ei olnud keegi kindel selle realiseerimise võimalikkuses. Selle tegemiseks analüüsiti esmalt olemasolevaid lahendusi ning otsiti võimalikult palju informatsiooni seda puudutavate teemade kohta. Rakenduse teostuskäik algas testrakenduse tegemisest, et tõestada rakenduse realiseerimise võimalikkust.

Teostamise käigus sõnastati ka rakenduse funktsionaalsed nõuded ja ootused rakenduse funktsionaalsuse kohta, millest kõik said rakenduse teostuse jooksul täidetud. Kirjeldati testrakenduse ning põhirakenduse arenduskäiku.

Lõputöö käigus tehti rakendus, mis oli väga uudne. Rakendus erines oma konkurentidest just kõige rohkem laulmise täpsuse poolest, mis see pakkus. Lisaks Androidi rakenduse valmimisele puutus autor kokku ka andmebaaside muutmisega, tagarakenduse arendusega ning palju muuga, millega ta varem ei olnud kokku puudunud.

Rakendus on oma funktsionaalsuselt lõppfaasis, kuid kasutajaliidese poolt saaks veel parandada. Selle töö kirjutamise hetkel on selle arendus seisma jäetud, et vaadata kas see saavutab populaarsuse kasutajate hulgas ning kas seda on mõtet edasi arendada. Rakendus on saadaval Google Play poes StarMe nime all.

Lõputöö käigus õppis autor väga palju uusi tehnoloogiaid, tutvus väga mitmete uute API-liidestega ning õppis rohkem kui alguses arvata oskas. Autor leiab, et antud rakenduse arendus oli tema jaoks igati positiivne ning on ka töö tulemusega väga rahul.

Kasutatud kirjandus

[1] „StarMe Talent koduleht,“ [Võrgumaterjal]. Available: https://starmetalent.com. [Kasutatud 30 4 2019].

[2] „StarMe Talent Instagram,“ [Võrgumaterjal]. Available: https://www.instagram.com/starmetalent. [Kasutatud 30 4 2019].

[3] „Picogram App,“ [Võrgumaterjal]. Available: https://play.google.com/store/apps/details?id=com.threembed.picogram.android&hl=en. [Kasutatud 29 4 2019].

[4] „Guitar Hero 5,“ [Võrgumaterjal]. Available: https://en.wikipedia.org/wiki/Guitar_Hero_5. [Kasutatud 29 4 2019].

[5] „Android Graphview,“ [Võrgumaterjal]. Available: https://github.com/jjoe64/GraphView. [Kasutatud 30 4 2019].

[6] „FFT explanaiton,“ [Võrgumaterjal]. Available: https://www.youtube.com/watch?v=spUNpyF58BY&t=3s. [Kasutatud 3 5 2019].

[7] „Pitch Detection Algorithm,“ [Võrgumaterjal]. Available: https://en.wikipedia.org/wiki/Pitch_detection_algorithm. [Kasutatud 3 5 2019].

[8] „TarsosDSP,“ [Võrgumaterjal]. Available: https://github.com/JorenSix/TarsosDSP. [Kasutatud 1 5 2019].

[9] H. K. Alain de Cheveigne, „YIN, a fundamental frequency estimator for speech and music,“ *The Journal of the Acoustical Society of America,* pp. 1917-1930, 2002.

[10] „Pitch Detection Methods (YinFFT),“ [Võrgumaterjal]. Available: https://aubio.org/doc/0.4.0/pitch_8h.html. [Kasutatud 4 5 2019].

[11] „LyricView,“ [Võrgumaterjal]. Available: https://github.com/markzhai/LyricView. [Kasutatud 1 5 2019].

[12] „Camera2Video,“ [Võrgumaterjal]. Available: https://github.com/googlesamples/android-Camera2Video. [Kasutatud 1 5 2019].

[13] „FFmpeg-Android,“ [Võrgumaterjal]. Available: https://github.com/bravobit/FFmpeg-Android. [Kasutatud 2 5 2019].

[14] „Java MP4 Parser,“ [Võrgumaterjal]. Available: https://github.com/sannies/mp4parser. [Kasutatud 2 5 2019].

[15] „Filezilla,“ [Võrgumaterjal]. Available: https://filezilla-project.org. [Kasutatud 2 5 2019].

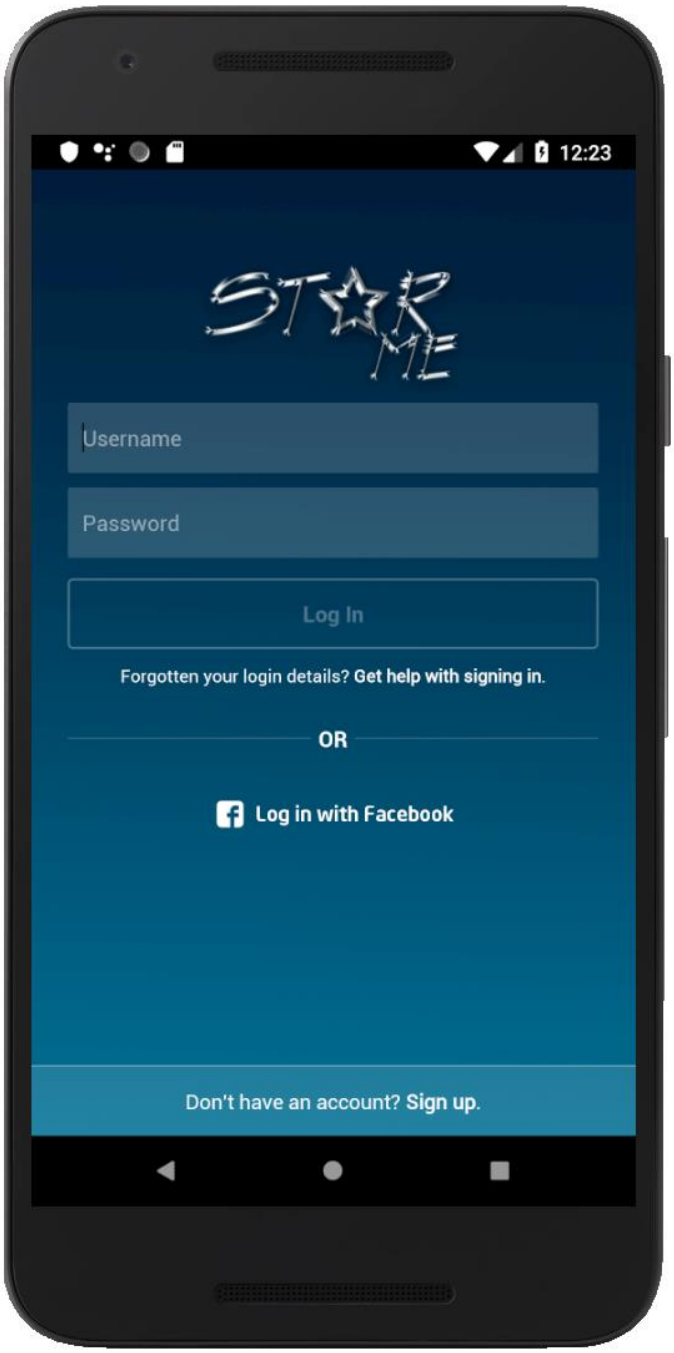
[16] „Neo4j,“ [Võrgumaterjal]. Available: https://neo4j.com. [Kasutatud 3 5 2019].

[17] „OkHttp,“ [Võrgumaterjal]. Available: https://square.github.io/okhttp. [Kasutatud 3 5 2019].

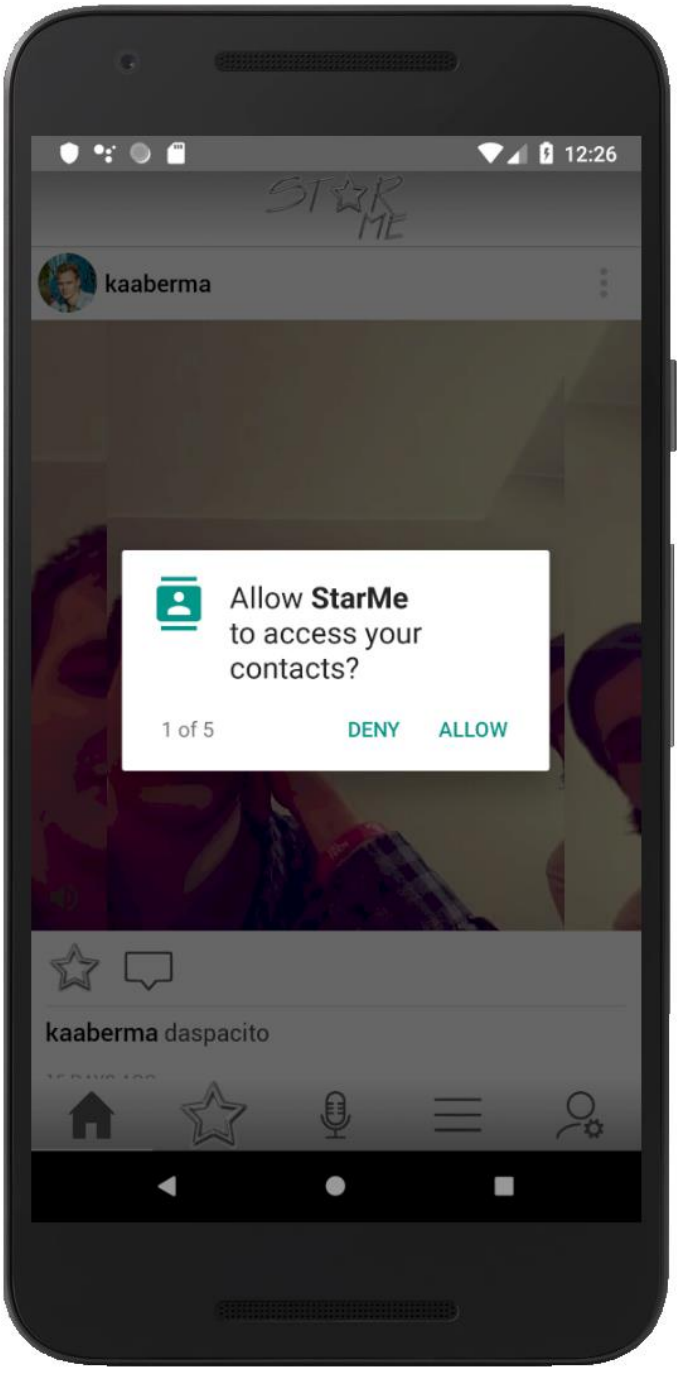
Lisa 1 – Kõik rakenduse kasutajaliidese vaated

Täpsustusena lisab autor, et vahetult enne piltide tegemist oli rakenduse andmebaasi tühjendatud, ehk rakenduses oli sellel hetkel ainult 1 postitus.

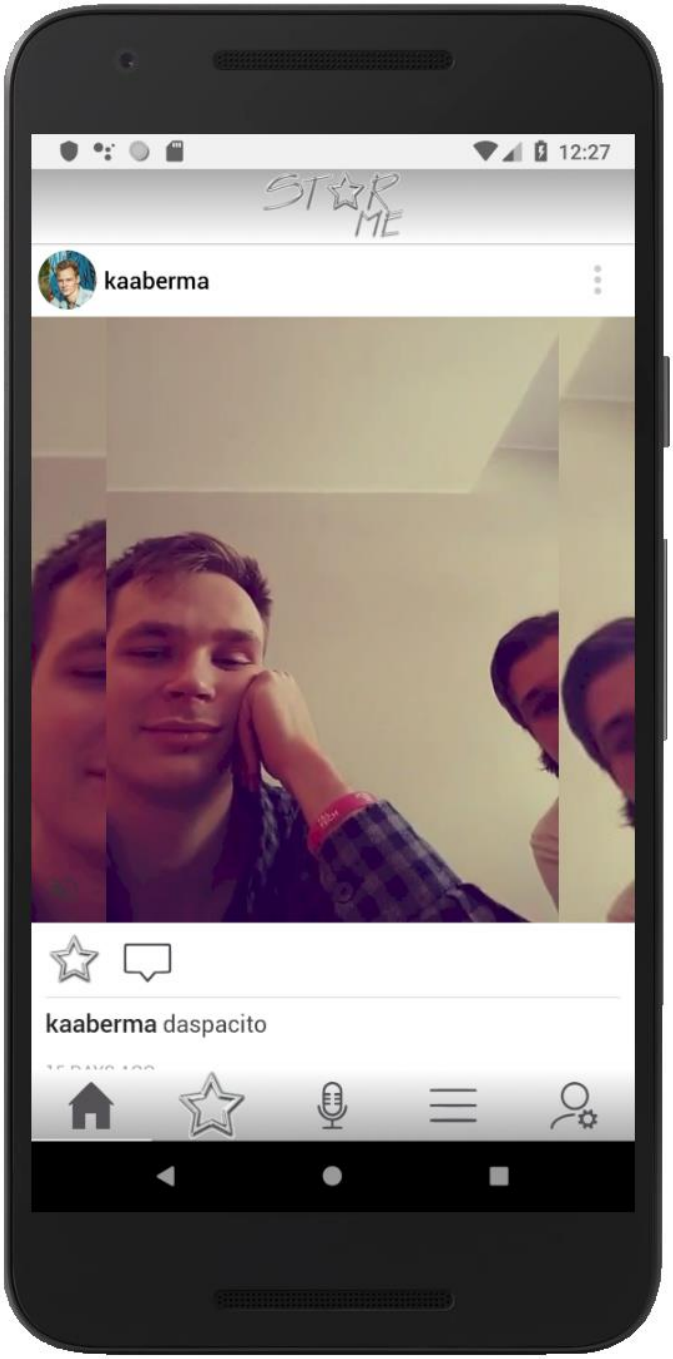
Sisselogimise vaade:



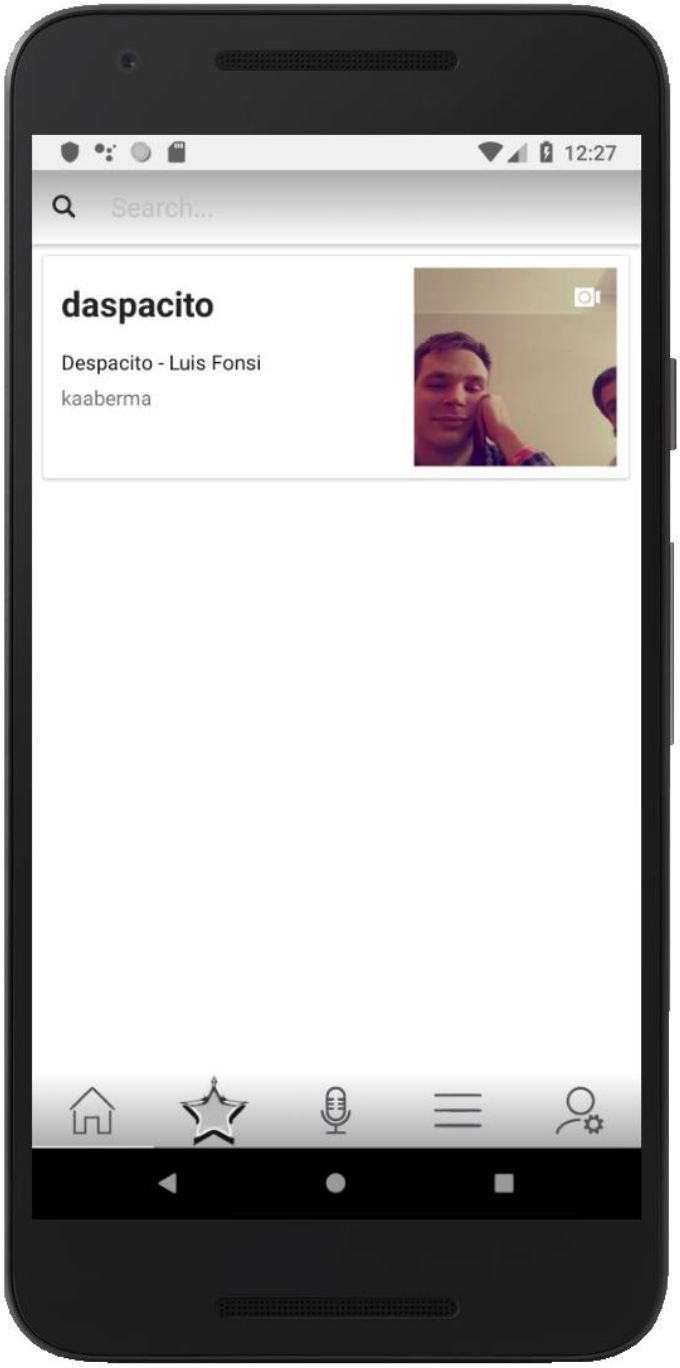
Esmasel sisselogimisel õiguste küsimine:



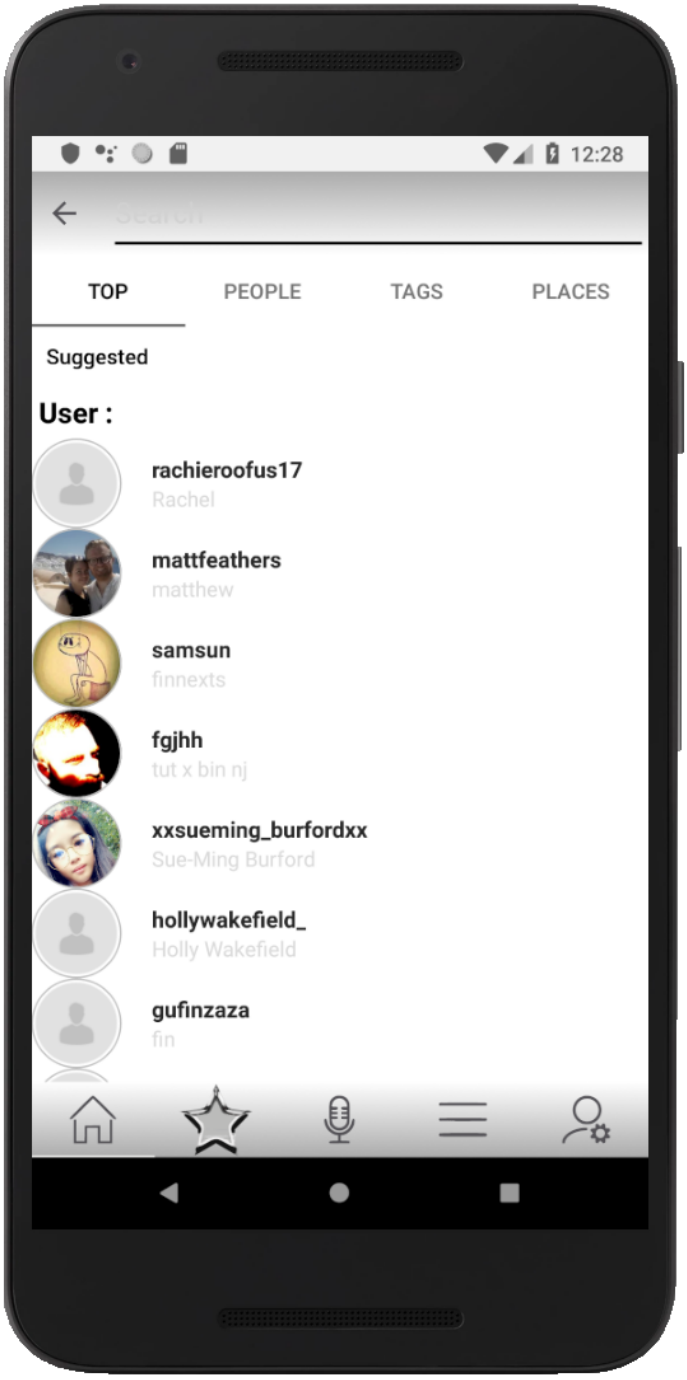
Avalehe vaade:



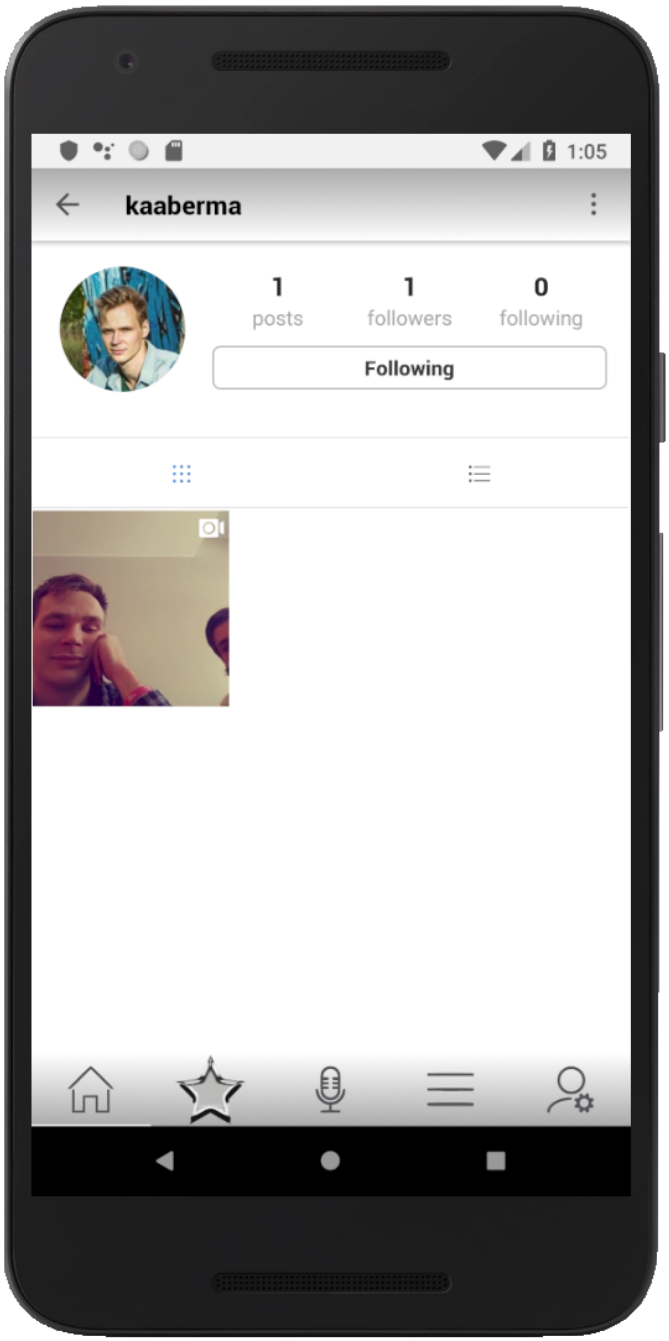
Kõikide andmebaasis olevate laulude vaade:



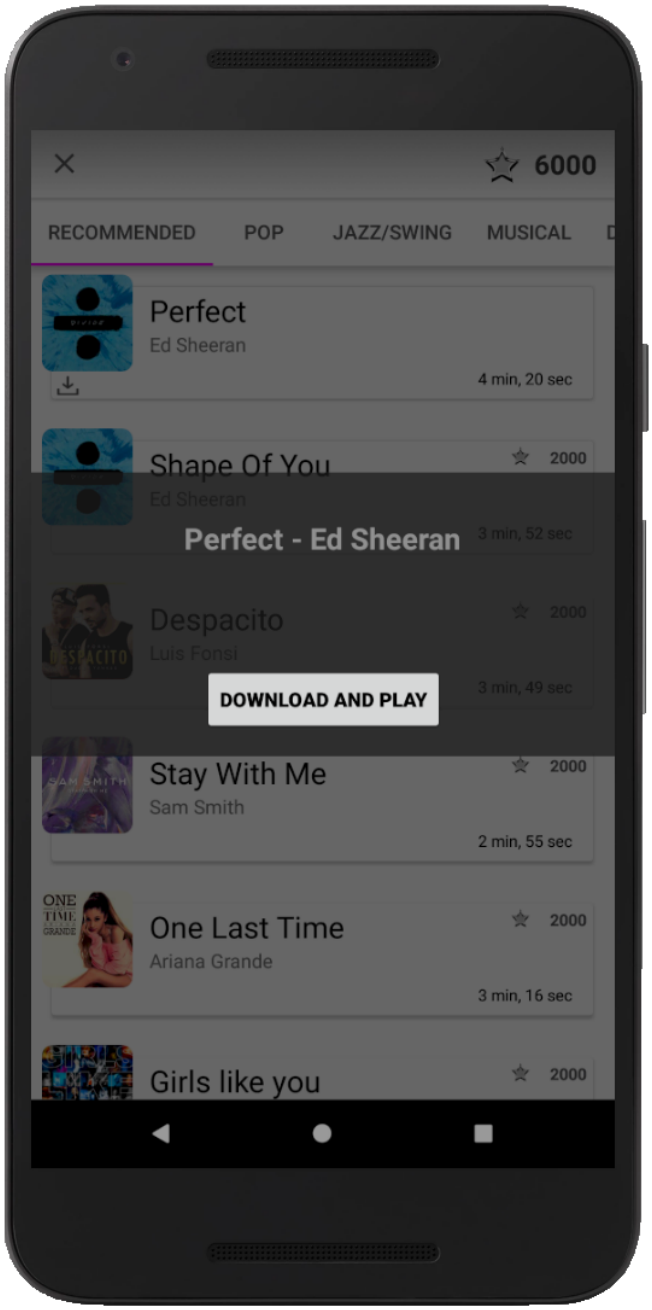
Nime järgi kasutaja otsimise vaade:



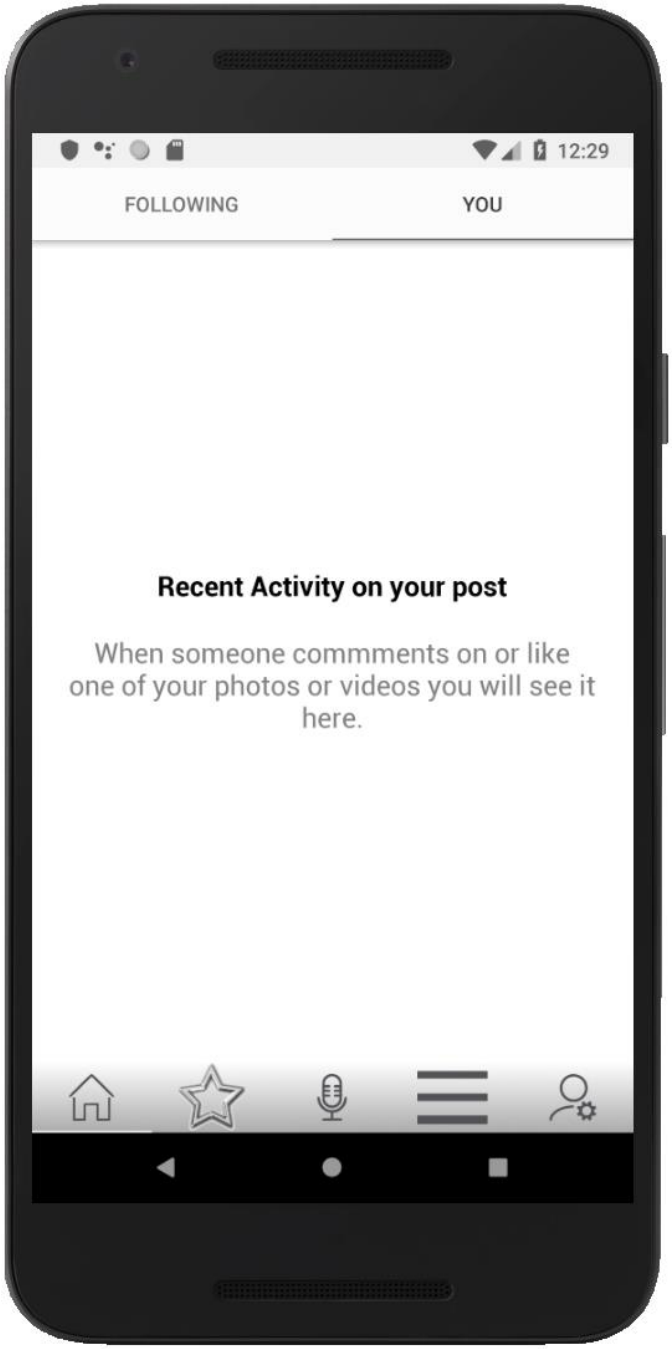
Teise kasutaja profiilivaade enda kasutaja alt vaadatuna:



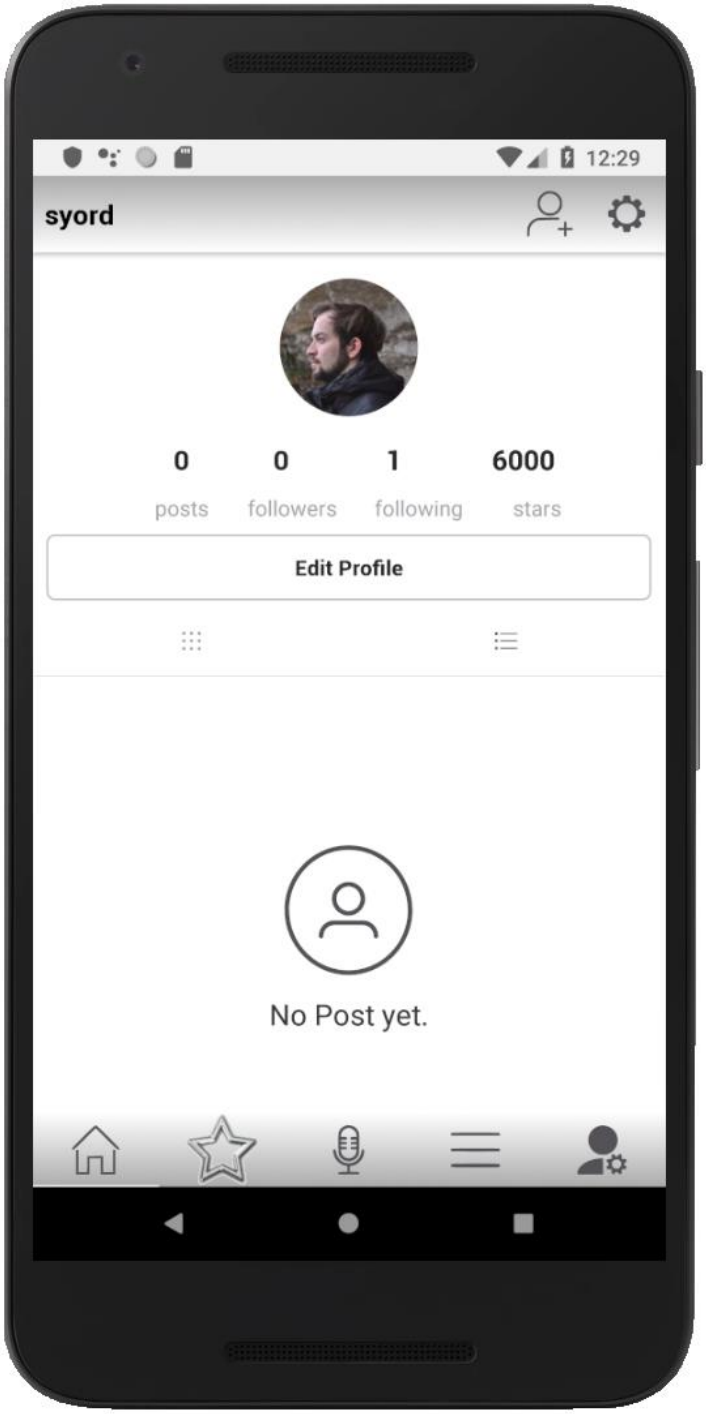
Kolmanda vaate laulu allalaadimise vaade:



Enda profiili tegevuste vaade:



Enda profiilivaade:



Seadete vaade: