



Ryhmä 23

Alexander San Miguel, Jani Peltonen, Jemila Matilainen, Kalle Österlund

MusicArch

Metropolia Ammattikorkeakoulu

Insinööri (AMK)

Tieto- ja viestintäteknologian tutkinto-ohjelma

Toteutusdokumentti

17.12.2021

Sisällys

Asiasanasto

| | |
|--|-----------|
| Johdanto | 1 |
| Käyttöohjeet | 1 |
| Aloitus | 1 |
| Etusivu | 1 |
| Yläpalkki | 3 |
| Kokoelma/Toivelista | 5 |
| Lisäyspyynnöt | 9 |
| Artistin lisäys | 10 |
| Genren lisäys | 10 |
| Albumin lisäys | 10 |
| Ohjeet | 12 |
| Tuotteen vaatimukset | 13 |
| Toiminnalliset vaatimukset ohjelmistolta | 13 |
| Asiakkaan ja ylläpitäjän yhteiset toiminnalliset vaatimukset | 13 |
| Ylläpitäjän omat toiminnalliset vaatimukset | 16 |
| Laadulliset vaatimukset ohjelmistolta | 18 |
| Käyttäjäroolit ja käyttötapaukset | 19 |
| Käsitteet, määritelmät ja ohjelmiston tietomalli | 22 |
| Ohjelmiston rakenne | 24 |
| Ohjelmiston toiminta | 29 |
| Kehitysprosessi ja kehitysvaiheen tekniikat | 32 |
| Ohjelmistotuotantoprojekti 1 | 32 |
| Ohjelmistotuotantoprojekti 2 | 34 |
| Testaus | 35 |
| Jatkokehitysmahdollisuudet | 35 |
| Yhteenveto | 36 |

Asiasanasto

Asiakas: Sovelluksen käyttäjä

Ylläpitäjä: Sovelluksen käyttäjä, joka voi lisätä sisältöä tietokantaan

Genre: Musiikin tyylilaji, johon kuuluu artisteja

Artisti: Musiikin tekijä tai yhtye, joka edustaa jotain genreä ja tekee albumeita

Albumi: Kokoelma kappaleita, joka edustaa yhtä tai useampaa genreä ja joka on merkitty yhdelle tai useammalle artistille

Kokoelma: Lista, jonne asiakas voi tallentaa levyjä

Toivelista: Lista, jonne asiakas lisää haluamansa albumit

Lisäyspyyntö: Asiakkaan pyyntö ylläpitäjälle, jonka tarkoituksena on ilmoittaa puutteista sovelluksen tietokannassa

Yläpalkki: Sovelluksen yläosan palkki, jonka avulla sovelluksessa navigoidaan.

1 Johdanto

Tämä dokumentaatio sisältää ohjelmistotuotantoprojekti 2- kurssilla toteutun ohjelmiston MusicArchin suunnittelun ja määrittelyn, toteutuspolun, rakenteen sekä käyttöohjeen. Dokumentaatio on suunnattu ohjelmistotuotannossa työskenteleville ammattilaisille avatakseen ohjelmiston rakennetta ja toiminnallisuutta, sekä mahdollistaakseen jatkokehityksen vaihtuvassa työympäristössä ja -tiimissä.

MusicArch on kokoelman hallintasovellus, jolla asiakas voi etsiä itseään kiinnostavaa musiikkia sovelluksen tietokannasta. Asiakkaalla on mahdollisuus tutustua uusiin genreihin, artisteihin ja albumeihin sovelluksessa. Albumeita voidaan etsiä sekä vapaalla tekstihaulla, että genre- tai artistikohtaisesti. Sovellus mahdollistaa käyttäjää lisäämään albumeita käyttäjän albumikokoelmaan tai -toivelistaan.

2 Käyttöohjeet

Aloitus

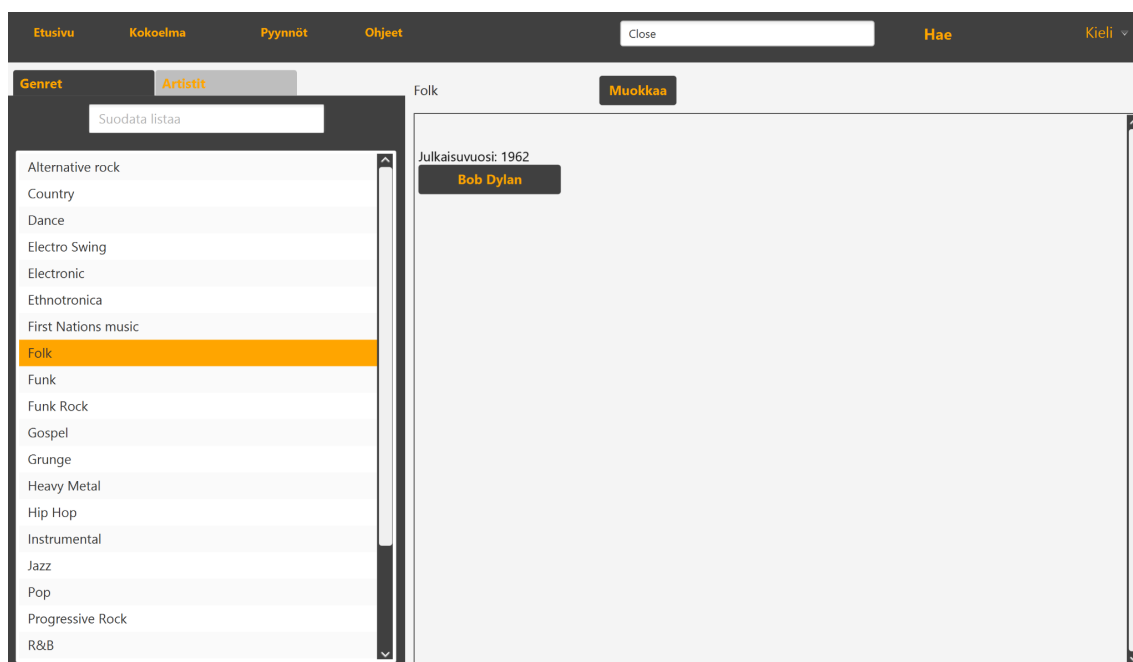
Tervetuloa käyttämään MusicArchia, kokoelman hallintasovellusta!

Sovelluksella on kaksi erilaista käyttäjäryhmää, asiakas ja ylläpitäjä. Ylläpitäjällä on pääsy sovelluksen Pyynnöt-ikkunaan, sekä mahdollisuus poistaa ja muokata genrejä, artisteja ja albumeita.

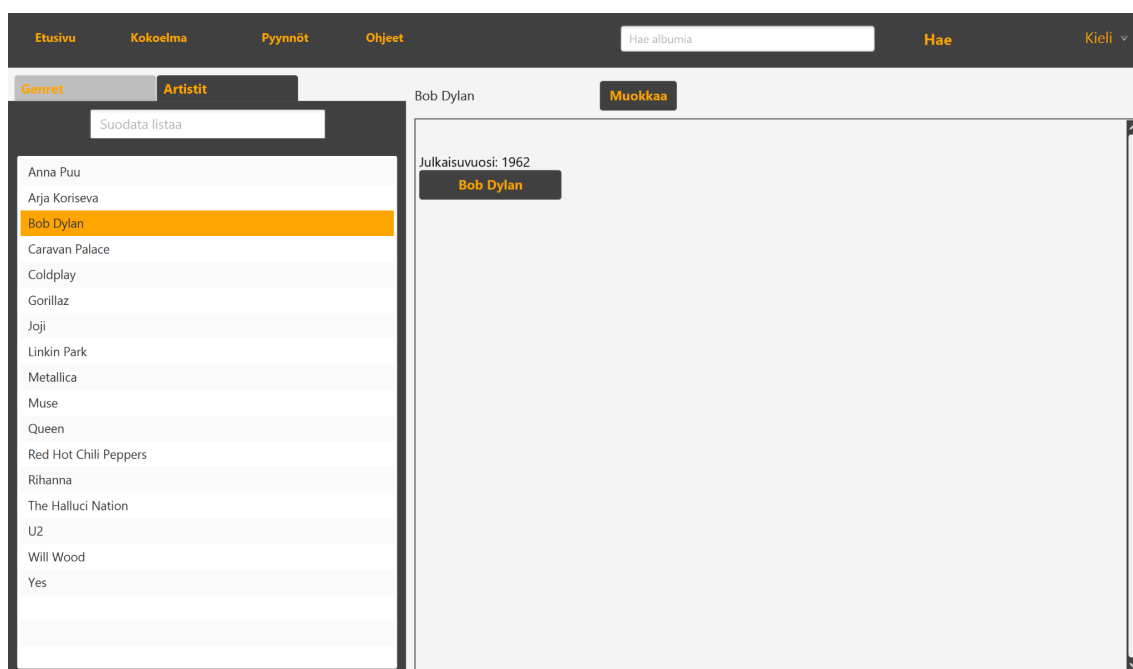
Kaikki käyttöohjeen kuvakaappaukset on ylläpitäjän sovelluksen näkymästä (muutokset ovat "Pyynnöt"-painike yläpalkissa ja "Muokkaa"-painikkeet genreissä, artisteissa ja albumisivuilla). Kerromme ohjeissa, mitkä ominaisuudet ovat ylläpitäjän käytössä.

Etusivu

Kun käynnistät sovelluksen asiakkaana päädyt etusivulle, missä voit tarkastella ja valita sovelluksessa olevia genrejä (kuva 1) ja artisteja (kuva 2) sekä näihin molempiin kuuluvia albumeita.



Kuva 1. Sovelluksen etusivun näkymä, jossa valintapaneelistä on valittu genre-välilehti.

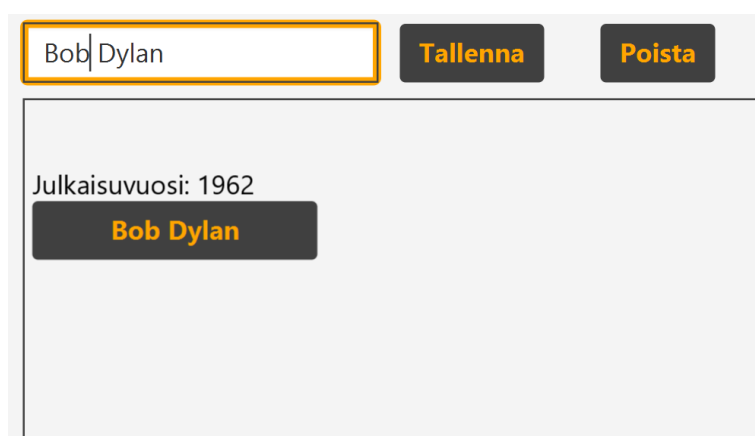


Kuva 2. Etusivun näkymä, jossa valintapaneelistä on valittu artistit-välilehti.

Ylläpitäjä voi myöskin muokata ja poistaa artisteja ja genrejä (kuva 3).

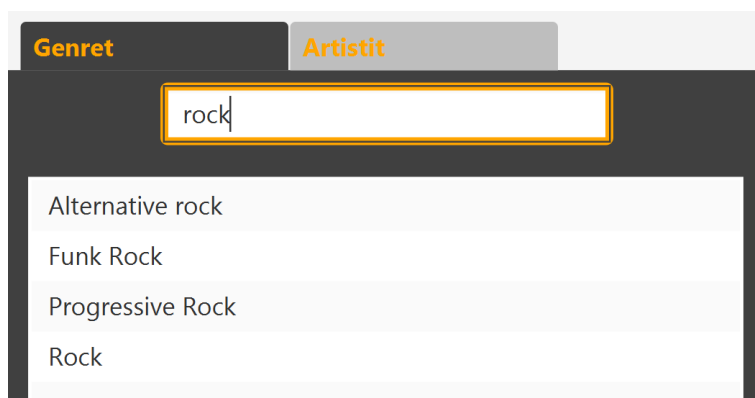
Painamalla muokkaus-painiketta voidaan muokata artistin nimeä tai poistaa artisti. Tämä ominaisuus toimii samalla tavalla genrejen suhteen.

Poistamalla artistin tai genren kaikki niihin liittyvät albumit myös poistetaan.



Kuva 3: Artistin muokkaus/poisto-toiminto.

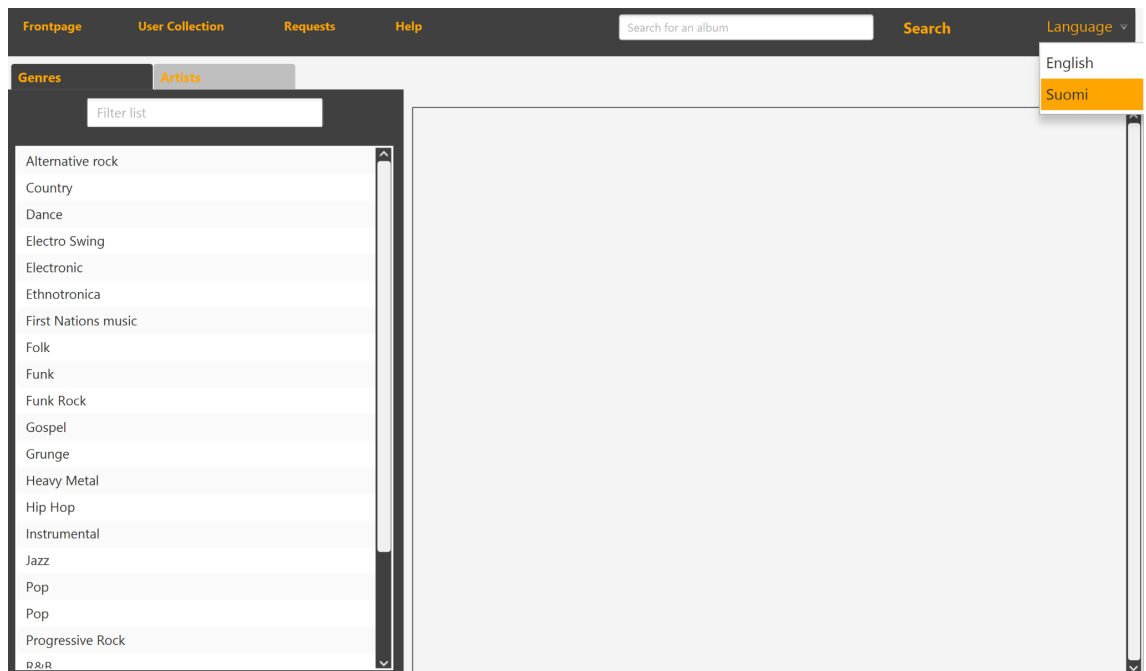
Vasemmalla voidaan etsiä genre- ja artistivälilehdeltä erilaisia genrejä ja artisteja (kuva 4). Hakutoiminto päivittää listan reaaliaikaisesti syötteen mukaiseksi, kun siihen kirjoitetaan.



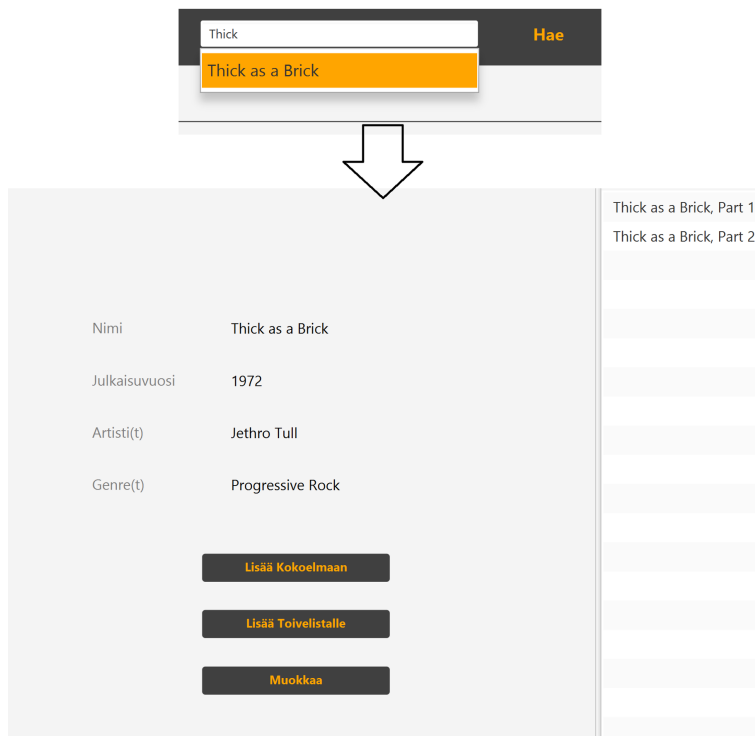
Kuva 4: Genrejen hakeminen listasta.

Yläpalkki

Yläpalkista pääset käsiksi sovelluksen muihin näkymiin (Kokoelma, Lisäyspyynnöt-ikkuna ylläpitäjälle ja ohjeet). Yläpalkki ja sen ominaisuudet ovat käytettävissä jokaisessa näkymässä, esimerkiksi hakutoiminto voidaan suorittaa albumin sivulta. Yläpalkissa voidaan myös suorittaa tekstihaku ja kielenvaihto (kuva 5). Haun avulla pääset suoraan albumin sivulle, jos hakemasi albumi on olemassa (kuva 6).



Kuva 5: Kielen valitseminen oikeasta ylänurkasta pudotusvalikosta.



Kuva 6: Yläpalkin haulla pääsee suoraan albumin sivulle.

Jos asiakas ei löydä sanahaun kautta etsittyä albumia, artistia tai genreä, hän voi lähettää lisäyspyynnön ylläpitäjille. (kuva 7).

Haun tulokset

Ei tuloksia artistilla
Nirvana

Ei tuloksia genrellä
Nirvana

Ei tuloksia albumilla
Nirvana

Kirjoita viesti

Missä on Nirvana?

Lisätkää Nirvana ja heidän
Nevermind albumi
tietokantaan|

57/250
kirjainta

Viestin otsikko

Lähetä pyyntö

Kuva 7: Lisäyspyynnön kirjoitus ja lähetys.

Kokoelma/Toivelista

Albumien sivuille pääsee käsiksi etusivulta tai suoraan hakemalla yläpalkista.

Kuvasta 6. voidaan huomata albumisivun kolme erilaista painiketta. Kahden ensimmäisen painikkeen avulla albumi voidaan lisätä asiakkaan/ylläpitäjän kokoelmaan tai toivelistalle.

Kolmas painike on vain ylläpitäjiä varten ja sillä saadaan muokattua albumin tietoja (nimi, julkaisuvuosi, artistien nimet sekä genrejen nimet) tai poistettua albumi sovelluksen tietokannasta (kuva 8). Albumin kappaleita ei voi muokata.

| | |
|---------------|---|
| Nimi | <input type="text" value="Thick as a Brick"/> |
| Julkaisuvuosi | <input type="text" value="1972"/> |
| Artisti(t) | <input type="text" value="Jethro Tull"/> |
| Genre(t) | <input type="text" value="Progressive Rock"/> |

Poista kokoelmasta

Lisää Toivelistalle

Tallenna

Poista albumi

Kuva 8: Albumin tietojen muokkaus.

Jos albumi lisätään kokoelmaan (kuva 9), se näytetään asiakkaan kokoelmasivulla. Kokoelmasivulla asiakas/ylläpitäjä voi tarkastella kokoelmaansa lisäämiään albumeita sekä poistaa niitä halutessaan. Albumille voidaan myös valinnaisesti kirjoittaa jokin kuvaus tai siihen liittyvä muistiinpano (kuva 10).

Jos albumi ollaan lisätty kokoelmaan, sitä ei voida enää lisätä toivelistalle ja jos albumi lisätään toivelistalle lisäyksen jälkeen kokoelmaan, se poistuu toivelistalta automaattisesti.

| | |
|---------------|------------------|
| Nimi | Thick as a Brick |
| Julkaisuvuosi | 1972 |
| Artisti(t) | Jethro Tull |
| Genre(t) | Progressive Rock |

Poista kokoelmasta

Lisää Toivelistalle

Muokkaa

Kuva 9: Albumi lisätty kokoelmaan.

| Albumit | Toivelista | Kokoelmasi | Kappaleet |
|------------------|------------|--|--|
| Thick as a Brick | | <p>Poista</p> <p>Nimi Thick as a Brick</p> <p>Vuosi 1972</p> <p>Artisti(t) Jethro Tull</p> <p>Genre(t) Progressive Rock</p> <p>Kuvaus 10/1000</p> <p>Hyvä levy!</p> <p>Lisää kuvaus</p> <p>Tallennettu</p> | Thick as a Brick, Part 1 Thick as a Brick, Part 2 |

Kuva 10: Albumi kokoelmassa. Kokoelmaan lisätyt albumit luetellaan Albumit-välilehden listanäkymässä.

Albumin voi myös lisätä toivelistalle (kuva 11) ja se voidaan poistaa sieltä manuaalisesti. Toivelistalle ei tallenneta muuta kuin albumin nimi (kuva 12).

| | | | |
|---------------|---------------------------|--------------|------------|
| Nimi | We Are the Halluci Nation | | |
| Julkaisuvuosi | 2016 | | |
| Artisti(t) | The Halluci Nation | | |
| Genre(t) | Dance | Ethnotronica | Electronic |

Lisää Kokoelmaan

Poista toivelistalta

Muokkaa

Kuva 11: Albumi lisätty toivelistalle.

Albumit Toivelista

We Are the Halluci Nation

Poista

Nimi

Vuosi

Artisti(t)

Genre(t)

Kuvaus

Lisää kuvaus

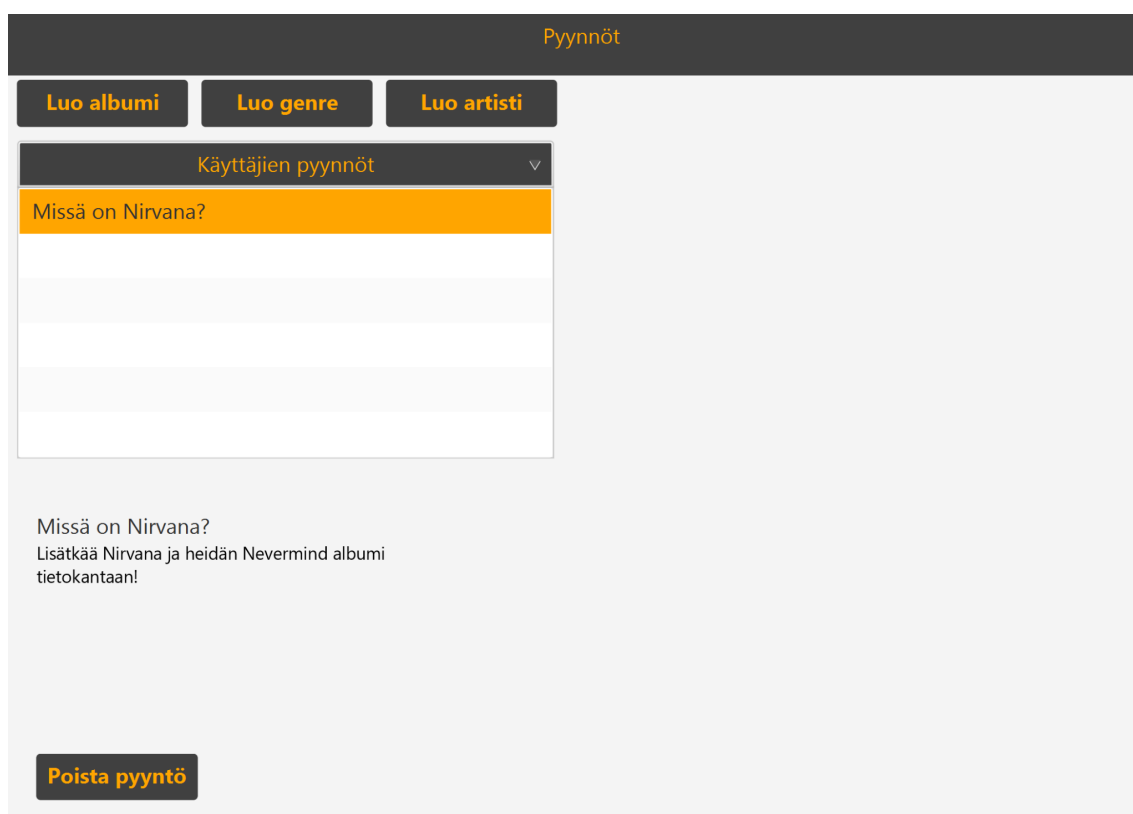
Kappaleet

Kuva 12: Albumi toivelistalla. Toivelistaan lisätyt albumit luetellaan Toivelista-välilehden listanäkymässä.

Lisäyspyynnöt

Vain ylläpitäjällä on pääsy lisäyspyyntöihin. Lisäyspyynnöt sijaitsevat erillisessä ikkunassa Pyyntö-painikkeen takana. Pyyntö-ikkunassa voidaan lisäksi lisätä puuttuvia genrejä, artisteja ja albumeita sovelluksen tietokantaan (kuva 13).

Ylläpitäjät voivat lukea asiakkaiden lähettämiä lisäyspyyntöjä, ja luoda niiden pohjalta uutta sisältöä sovelluksen tietokantaan (kuva 13).



Pyyntö

Luo albumi Luo genre Luo artisti

Käyttäjien pyynnot

Missä on Nirvana?

Missä on Nirvana?
Lisätkää Nirvana ja heidän Nevermind albumi tietokantaan!


Poista pyyntö

Kuva 13: Pyyntö-ikkuna, sekä kuvassa 7 nähty asiakkaan lisäyspyyntö.

Artistien, genrejen ja albumien lisäys-sivuille pääsee pyyntöjen yläpuolella olevilla painikkeilla. Jos genre, artisti tai albumi löytyy jo tietokannasta, siitä ilmoitetaan lisäyksessä. Et voi lisätä olemassa olevia artisteja, genrejä tai albumeita sovelluksen tietokantaan.

Artistin lisäys

Artistille annetaan nimi (pakollinen), sekä biografia (valinnainen). Sen jälkeen artistin lisäys lähetetään sovelluksen tietokantaan (kuva 14).



Lisää artisti

Nimi

Biografia

Grunge band from Seattle.

Lähetä

Kuva 14: Artistin lisäys.

Genren lisäys

Genrelle annetaan vain nimi (pakollinen). Sen jälkeen genren lisäys lähetetään sovelluksen tietokantaan (kuva 15).



Lisää genre

Nimi

Lähetä

Kuva 15: Genren lisäys.

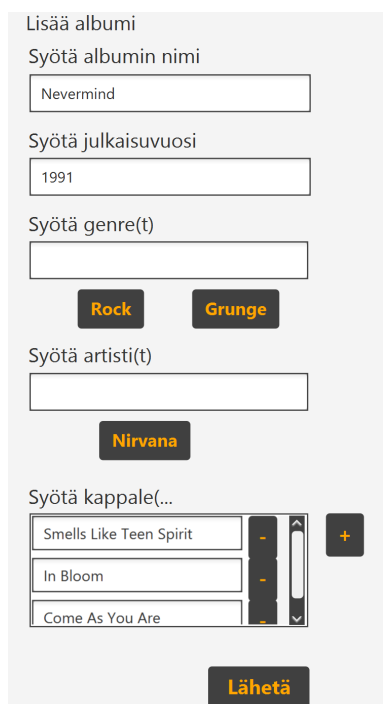
Albumin lisäys

Albumille annetaan albumin nimi (pakollinen), julkaisuvuosi (pakollinen, vuosi neljänä numerona, ei päivämääriä), artistien ja genrejen nimet (jos artisteja tai

genrejä on useampi, niitä voidaan lisätä albumille. Molempia saa lisätä maksimissaan kolme kappaletta).

Olemassa olevat genret sekä artistit tulevat esiin automaattisesti esille kirjoitettaessa syötettä ja ne voidaan merkitä albumille klikkaamalla haluttua hakutulosta (kuva 17). Toiminto on samanlainen artistien kanssa.

Albumiin voidaan lisätä myös kappaleita (valinnaista). Tämän jälkeen albumin lisäys saadaan lähetettyä sovellukseen ja se lisätään tietokantaan (kuva 16).



Lisää albumi

Syötä albumin nimi

Nevermind

Syötä julkaisuvuosi

1991

Syötä genre(t)

Rock Grunge

Syötä artisti(t)

Nirvana

Syötä kappale(...)

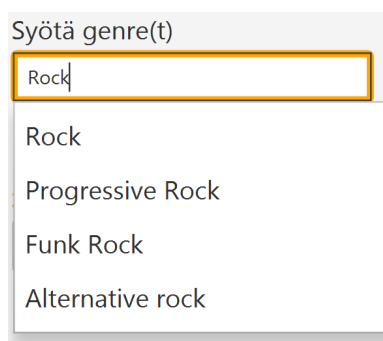
Smells Like Teen Spirit - +

In Bloom -

Come As You Are -

Lähetä

Kuva 16. Albumin lisäys.



Syötä genre(t)

Rock

Rock

Progressive Rock

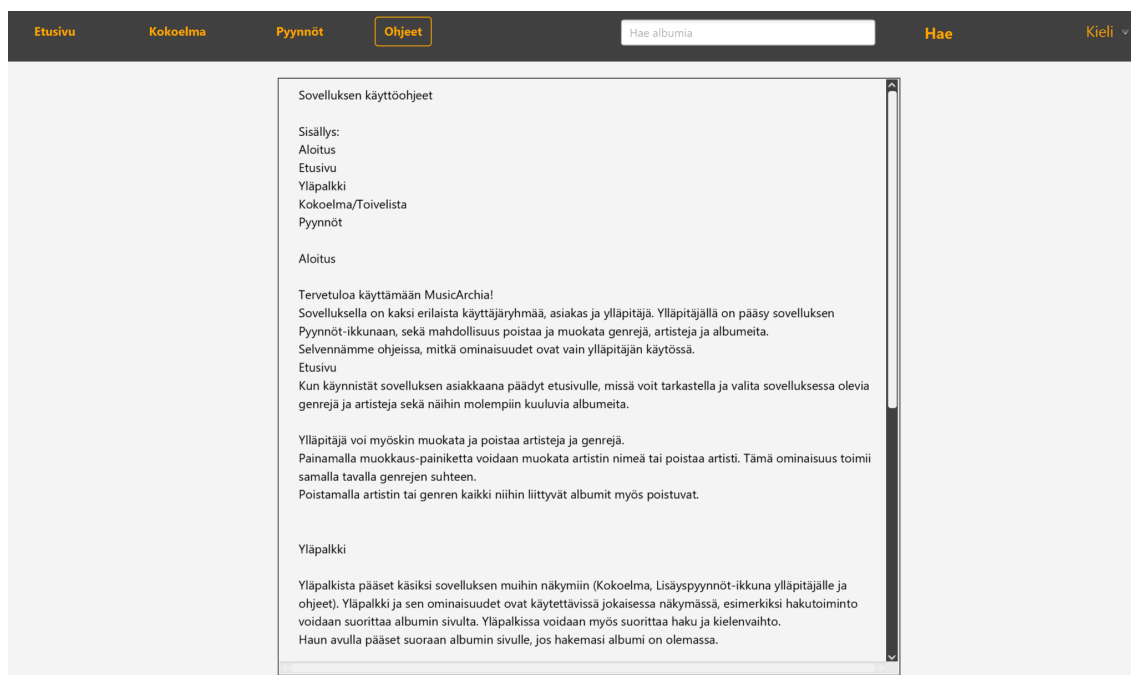
Funk Rock

Alternative rock

Kuva 17: Albumin genrejen lisäys.

Ohjeet

Ohje-sivulta voit lukea sovelluksen käyttöohjeet tekstiversiona (kuva 18).



Kuva 18. Ohjesivun näkymä.

3 Tuotteen vaatimukset

Tekninen dokumentaatio alkaa tästä.

Tuotteen visio: mitä ja miksi?

Projektin tavoitteena on luoda musiikkiaineistolle tietopankki, josta käyttäjän on helppo etsiä ja tarkastella eri genrejen alta artisteja, albumeita ja kappaleita. Käyttäjällä on myös mahdollisuus vaikuttaa olemassa olevaan tietoon ja tuottaa uutta tietoa lähettämällä lisäys- tai muokkauspyyntöjä ylläpitäjälle. Artisteilta käyttäjä voi tarkastella lyhyttä biografiaa ja levyistä julkaisuvuotta. Käyttäjä pystyy luomaan omia levykokoelmia ja arvostelemaan levyjä antaen arvosanan 1-5 ja kirjoittaen vaihtoehtoisesti jonkin arvostelun.

Musiikkikirjasto kokoaa tiedon helposti hallittavaan ja käytettävään pakettiin, jolloin vältetään tiedon hajanaisuus ja etsimisen hitaus.

MusicArch on standalone-pöytäkonesovellus, joka toimii käyttäjien päätelaitteella. Tämän avulla, jos palvelin on kaatunut, käyttäjä voi silti katsella lokaalisti tallennettuja listoja, joka ei perinteisesti käytetyllä web-sovelluksella olisi mahdollista.

Toiminnalliset vaatimukset ohjelmistolta

Asiakkaan ja ylläpitäjän yhteiset toiminnalliset vaatimukset

Asiakas pystyy etsimään hakusanoilla albumeita.

Järjestelmän tulee hakea etätietokannasta **Album-taulusta** tietoa käyttäjän asettamalla hakusyötteellä.

Asiakas voi selata hakutuloksia

Järjestelmän tulee listata etätietokannasta tuodut hakutulokset kategorioittain (genre, artisti, albumi) käyttäjälle selkeällä tavalla.

Asiakas voi tarkastella artistien tuotantoa.

Järjestelmän tulee tuoda etätietokannasta **Artist**-taulusta asiakkaan valitseman artistin nimen ja tuotannon, eli albumilistan käyttäjälle luettavassa muodossa.

Asiakas voi tarkastella albumeita, niiden julkaisuvuosia ja kappaleita.

Järjestelmän tulee hakea asiakkaan valitseman albumin tiedot etätietokannasta **Album**-, **AlbumArtists**-, **AlbumGenres**- ja **Song**-tauluista. Albumin tiedot sisältävät albumin nimen, biografian, artistit, genret, julkaisuvuoden sekä kappaleet.

Asiakas voi lisätä albumeita kokoelmaan tai toivelistaan.

Järjestelmän tulee hakea asiakkaan valitseman albumin tiedot etätietokannasta **Album**-, **AlbumArtists**-, **AlbumGenres**- ja **Song**-tauluista sekä tallentaa ne asiakkaan paikalliseen tietokantaan.

Asiakas voi arvioida oman kokoelmansa albumeita.

Järjestelmän tulee tallentaa paikalliseen tietokantaan albumikohtaisesti käyttäjän kirjoittaman arvioinnin albumista.

Asiakas voi tarkastella albumien tietoja

Järjestelmän tulee hakea etätietokannasta albumien tiedot **Album-** ja **Song-**tauluista käyttäen **AlbumSongs-**välitaulua ja näyttää ne käyttöliittymässä.

Asiakas voi pitää listaa omista levyistään kokoelmassa

Asiakkaan valitseman albumin tiedot haetaan etätietokannasta **Album-** ja **Song-** tauluista käyttäen **AlbumSongs-** välitaulua ja tallennetaan lokaaliin tietokantaan **Album-** ja **Songs-**tauluihin käyttäen **AlbumSongs-** välitaulua.

Asiakas voi poistaa albumin kokoelmastaan tai toivealbumilistastaan

Järjestelmän tulee hakea asiakkaan valitseman albumin tiedot paikallisesta tietokannasta **Album-**, **AlbumArtists-**, **AlbumGenres-** ja **Song-**tauluista sekä poistaa ne.

Asiakas voi tarkastella kokoelman albumien tietoja

Järjestelmän tulee hakea paikallisesta tietokannasta albumien tiedot **Album-** ja **Song-**tauluista käyttäen **AlbumSongs-**välitaulua ja näyttää ne käyttöliittymässä.

Asiakkaana haluan käyttää ohjelmaa omalla äidinkielelläni

Järjestelmän tulee vaihtaa sovelluksen kieli suomeksi tai englanniksi, käyttäen

Asiakkaana haluan lähettää tietueen lisäyspyyntöjä

Asiakkaan tulee olla mahdollista lähettää vapaamuotoinen pyyntö uuden tietueen lisäystä varten, mikäli hän havaitsee puuttuvan Genren, Artistin tai Albumin.

Ylläpitäjän omat toiminnalliset vaatimukset

Ylläpitäjä voi lisätä genrejä

Järjestelmän tulee tarkistaa löytyykö lomakkeeseen syötetty genren nimi etätietokannasta, jolloin löytyessä uutta genreä ei lisätä. Jos genren nimeä ei löydy lisää järjestelmä etätietokantaan uuden genren **Genre**-tauluun genrenimen perusteella ja antaa tälle automaattisen id-numeron.

Ylläpitäjä voi lisätä artisteja ja artistikohtaisia biografioita.

Järjestelmän tulee tarkistaa löytyykö lomakkeeseen syötetty artistin nimi etätietokannasta, jolloin tiedon löytyessä uutta artistia ei lisätä. Jos artistin nimeä ei löydy lisää järjestelmä etätietokantaan **Artist**-tauluun uuden artistin nimen ja lisää käyttäjän syöttämän biografian ja antaa datalle automaattisen id-numeron.

Ylläpitäjä voi lisätä albumeita ja antaa niille julkaisuvuoden, genret, artistit ja kappaleet.

Järjestelmän tulee tarkistaa löytyykö lomakkeeseen syötetty albumin nimi etätietokannasta, jolloin löytynyttä tietoa uutta albumia ei lisätä tai muita tietoja ei lisätä. Jos albumin nimeä ei löydy asettaa järjestelmä etätietokantaan uuden albumin nimen ja julkaisuvuoden **Album**-tauluun, sekä luo automaattisen id-numeron. Järjestelmä tarkistaa löytyvätkö syötetyt artistinimet ja genrenimet etätietokannasta, jolloin löytyvät artistit lisään id-numeron perusteella ja ei-löytyvät jätetään lisäämättä kokonaan. Sovellus jatkaa tietojen tallentamista asettamalla ylläpidon yhdistämille tietueille automaattisen artistin id- numeron ja albumin id-numeron. **AlbumArtists**-tauluun asetetaan Albumin id-numero artistin id-numeron kanssa. **AlbumGenres**-tauluun tallennetaan ylläpidon syöttämien genrejen id-numerot albumin id-numeron kanssa.

Ylläpitäjänä haluan muokata genrejä

Järjestelmän tulee ottaa talteen ylläpitäjän syöttämän tiedon ja muokattavan genren id-numeron. Uusi genrenimi tallennetaan id-numeron avulla vanhan genrenimen tilalle etätietokantaan **Genre**-tauluun.

Ylläpitäjänä haluan muokata artisteja

Järjestelmän tulee ottaa talteen ylläpitäjän syöttämän tiedon ja muokattavan artistin id-numeron. Uusi artistinimi tallennetaan id-numeron avulla vanhan artistinimen tilalle etätietokantaan **Artist**-tauluun.

Ylläpitäjänä haluan poistaa albumeita ja kappaleita tietokannasta

Järjestelmän tulee hakea ylläpitäjän valitseman albumin tiedot etätietokannasta **Album**-, **AlbumArtists**-, **AlbumGenres**- ja **Song**-tauluista sekä poistaa ne.

Ylläpitäjänä haluan muokata albumeita tietokannassa

Järjestelmän tulee ottaa talteen ylläpitäjän syöttämän tiedot ja muokattavan albumin id-numeron. Järjestelmä korvaa id-numeron avulla etätietokantaan uuden albumin nimen ja uuden julkaisuvuoden **Album**-tauluun. Järjestelmä tarkistaa löytyvätkö syötetyt artistinimet ja genrenimet etätietokannasta, jos artistinimi ja/tai genrenimi löytyvät, artistin nimi ja/tai genren nimi vaihdetaan id-numeron perusteella uusiin nimiin. Jos syötettyä tietoa (genren nimi tai artisin nimi) ei löyty tietokannasta, estää järjestelmä lisäämisen. Sovellus jatkaa tietojen tallentamista asettamalla **AlbumArtists**-tauluun albumin id-numeron uuden artistin id-numeron kanssa. **AlbumGenres**-tauluun tallennetaan ylläpidon syöttämien uusien genrejen id-numerot albumin id-numeron kanssa.

Ylläpitäjänä haluan selata asiakkaiden lähettämiä lisäyspyyntöjä

Järjestelmän tulee hakea etätietokannasta **UserRequests**-taulusta sinne tallennetut viestit ja niiden otsikot ja tuoda ne käyttöliittymään ylläpitäjän selattavaksi.

Laadulliset vaatimukset ohjelmistolta

MusicArchin pyrkimyksenä on luoda käyttäjälle selkeä ja helppokäyttöinen käyttöliittymä, jossa asiakkaan on helppo seurata ja ohjata toimintaansa. Käyttöliittymä toteutettiin responsiivisena, jotta datan lisääntyessä käyttäjän kuormitus ei kasvaisi ja käyttökokemus kärsisi.

Ohjelmiston toiminnallisuudessa pyrittiin ottamaan koko ajan huomioon kommunikaation käyttäjän ja ohjelmiston välillä: ohjelmisto vastaa käyttäjän tekemiin komentoihin kertoen mitä ohjelmassa tapahtuu tai on tapahtumassa

ajantasaisesti. Näin voidaan varmistaa, että käyttäjältä ei jää tapahtumia huomaamatta ja järjestelmän toiminta ei tunnu käyttäjästä epäjohdonmuokaiselta.

Virhetilanteet on otettu huomioon suunnittelussa ja toteutuksessa. Järjestelmä on suunniteltu sietämään virhetilanteita kuten virheellisiä hakuja syötteitä (esimerkiksi kirjoitusvirheet) ehdottaen syötteitä eniten vastaavia todennäköisimpiä hakutuloksia, jolloin käyttäjä saa aina haulle ehdotuksia. Myöskin ylläpitäjän lisätessä genre-, artisti- tai albumidataa tietopankkiin, varmistaa ohjelma haluaako käyttäjä varmasti lisätä syötetyt tiedot, sekä tarkistaa löytyvätkö vastaavat tiedot jo ennestään etätietokannasta.

Tietoturva otetaan huomioon sekä suunnittelussa, että kehityksessä. Käyttäjistä ei tallenneta etätietokantaan minkäänlaisia käyttäjätietoja, vaan käyttäjän lisäämät albumit ja niiden arvioinnit tallennetaan käyttäjän omaan henkilökohtaiseen paikalliseen tietokantaan. Paikalliseen tietokantaan tallennettava data ei sisällä käyttäjästä henkilökohtaista tietoa. Ohjelmisto ladataan käyttäjälle yksityiseen käyttöön henkilökohtaiselle koneelle. Ylläpitäjällä tai asiakkaalla ei ole mahdollista päästä muokkaamaan tietokannan rakennetta.

Järjestelmää suunniteltaessa otetaan huomioon laajennettavuus ja jatkokehitysmahdollisuudet. Tavoitteena on, että MusicArch- tietokanta joustaisi tarvittaessa tietokannan laajentuessa taikka datan määrän kasvaessa. Jatkokehitys kielen vaihtojen mahdollistamiseksi on huomioitu datan tallennuksessa, joten esimerkiksi kovakoodausta vältetään, jotta tietoa olisi mahdollisimman helppo muunnella.

4 Käyttäjäroolit ja käyttötapaukset

Käyttäjäroolit

Sovelluksessa on kaksi käyttäjäroolia, asiakas ja ylläpitäjä. Molemmat voivat käyttää sovellusta samanlaisin tavoin, mutta ylläpitäjällä on ylimääräisiä ominaisuuksia.

Ylläpitäjä huolehtii tietojen lisäämisestä, muokkaamisesta ja poistamisesta.

Asiakkaan käyttötapaukset

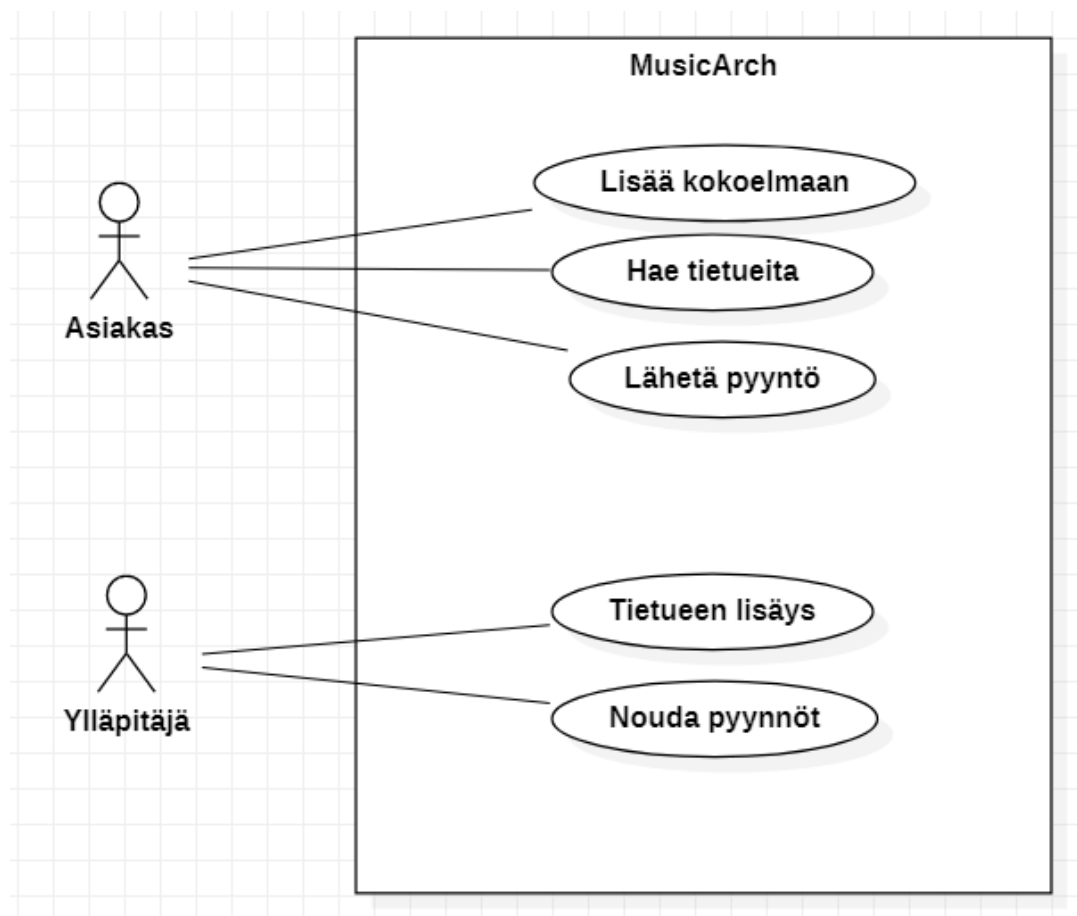
Asiakas voi selata sovelluksesta genrejä, artisteja ja niihin liittyviä albumeita. Hän voi lisätä albumeita omaan kokoelmaansa tai toivelistalleen. Hän voi etsiä albumeita sovelluksen hausta. Jos etsittävää tietoa ei löydy, hän voi lähettää lisäyspyynnön ylläpitäjille. Asiakas voi kirjoittaa kokoelmassaan kuvauksia albumeille.

Asiakas voi etsiä haluamiaan albumeita järjestelmästä tekstisyytteellä. Jos asiakas ei löydä etsimäänsä genreä, artistia tai albumia, niin hän voi lähettää ylläpidolle pyynnön haluamistaan tiedoista. Hän voi lisätä haluamansa albumin henkilökohtaiseen kokoelmaan. Hän voi kirjoittaa kokoelmaan lisättyyn albumiin oman henkilökohtaisen arvion tai kommentin albumista.

Ylläpitäjän käyttötapaukset

Ylläpitäjä voi asiakkaan ominaisuuksien lisäksi lisätä, muokata ja poistaa artisteja, genrejä ja albumeita. Hän voi lisätä artistien nimet ja biografiat sekä genrejen nimet tietokantaan. Albumeille lisätään nimi, julkaisuvuosi, artisti(t), genre(t) ja kappaleet. Albumin artistien ja genrejen on oltava ennestään tietokannassa jos albumi halutaan lisätä. Olemassaolevaa tietoa ei voida lisätä tietokantaan.

Hän voi myöskin tarkastaa asiakkaan lähettämät lisäyspyynnöt ja lisätä niiden pohjalta sovellukseen sisältöä.



Käyttötapauskaavio.

5 Käsitteet, määritelmät ja ohjelmiston tietomalli

Ohjelmalla on kaksi käyttäjäryhmää: asiakas ja ylläpitäjä. Asiakas on ohjelmistoa käyttävä henkilö, joka haluaa hallinnoida virtuaalista albumikokoelmaa ohjelmistossa. Ylläpito vastaanottaa asiakkaan albumi-, artisti- ja genrepyyntöjä, sekä lisää niitä tietokantaan.

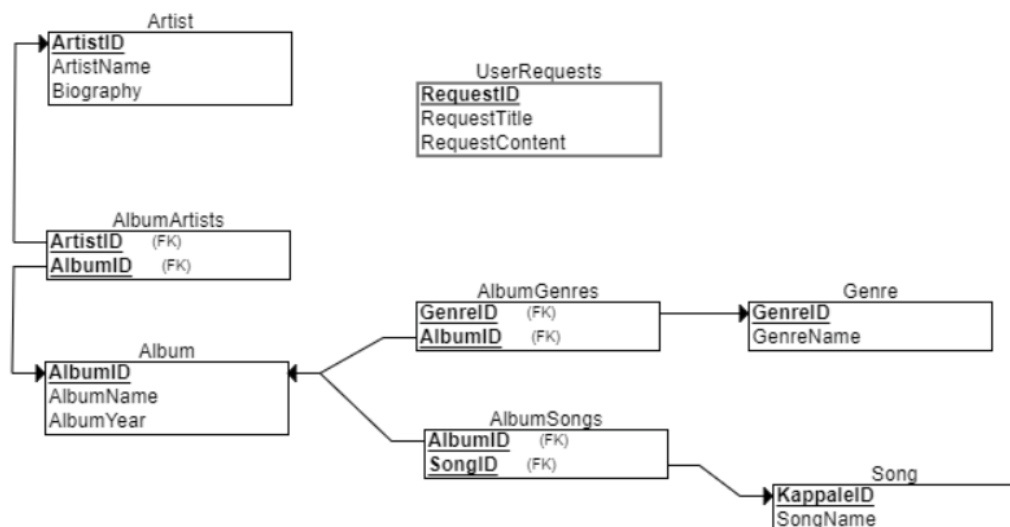
Albumi on fyysiseen kokoelmaan verrattavissa oleva termi, joka voi olla esimerkiksi vinyyllilevy, c-kasetti tai cd. Albumilla on vähintään yksi artisti ja

genre, sekä vähintään yksi kappale. Albumin tietueella artisti, voidaan etsiä yhdistettyjä albumeita. Artisti on albumin tekijä, kuten bändi, yksittäinen artisti tai vieraileva artisti. Albumiin kuuluvalla tietueella genre, voidaan etsiä myös yhdistettyjä albumeita.

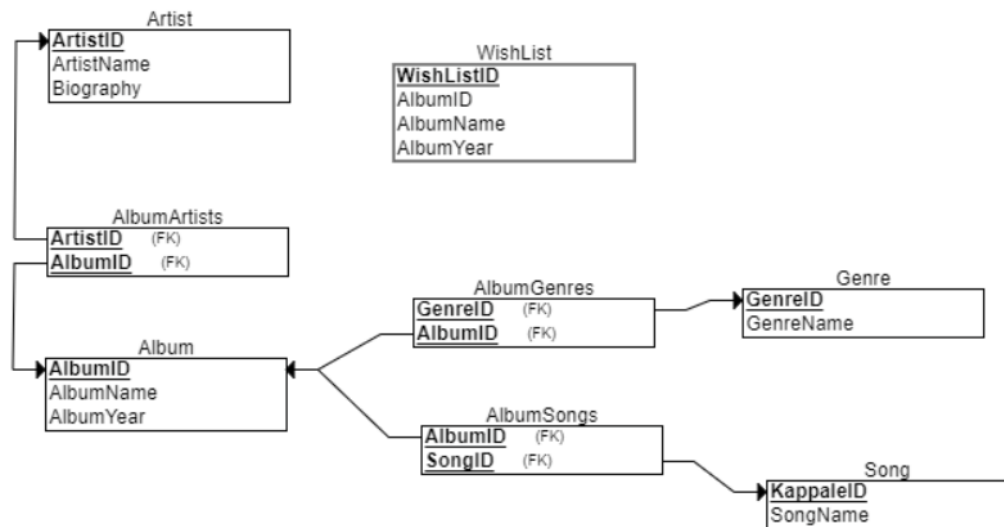
Albumiin kuuluu myös kappaleita, jotka ovat omassa välitaulussaan. Kappaleet eivät voi olla olemassa itsenäisesti vaan ne tarvitsevat aina albumiin liitoksen.

UserRequests on asiakkaan lähettämä lisäyspyyntö-olio, joka sisältää asiakkaan määrittelemän otsikon ja viestikentän. Ylläpito pääsee lisäyspyyntöihin käsiksi käyttöliittymän kautta, ja voi tarkastella olion tietoja sekä poistaa haluamiaan olioita kun pyydetty tietue on lisätty.

WishList on asiakkaan toivoman albumin tiedot sisältävä olio. Tämä tallennetaan paikalliseen tietokantaan, jotta se on asiakkaalle aina olemassa.



Kuvassa esitys etätietokannan rakenteesta.



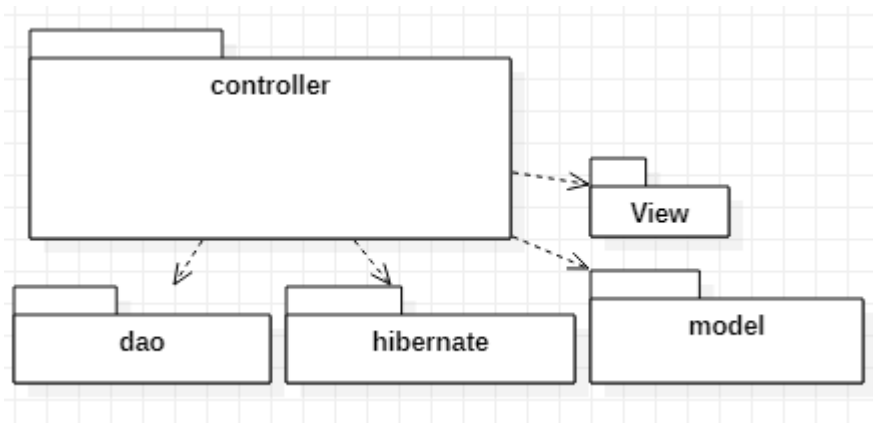
Kuvassa esitys asiakkaan lokaalin tietokannan rakenteesta.

6 Ohjelmiston rakenne

MusicArch tullaan jakamaan kahteen sovellukseen. Asiakkaille annettava sovellus, jolla hallinnoidaan omaa paikallista tietokantaa, joka pohjautuu etätietokantaan, sekä ylläpidon sovellus, jolla luodaan tietueita samaan tietokantaan.

Joka tapauksessa, asiakas asentaa itselleen sovelluksen, jossa ei ole etätietokantaan lisäämisominaisuutta. Ylläpidolle asennetaan sovellus, jossa voi lisäksi lisätä tietueita etätietokantaan. Molemmat sovellukset sisältävät oman MariaDB tietokantansa. Etätietokanta tällä hetkellä ajetaan Metropolian Educloudin sisäverkossa, joka sisältää opiskelijan virtuaalikoneen, johon on myös asennettu MariaDB-palvelu.

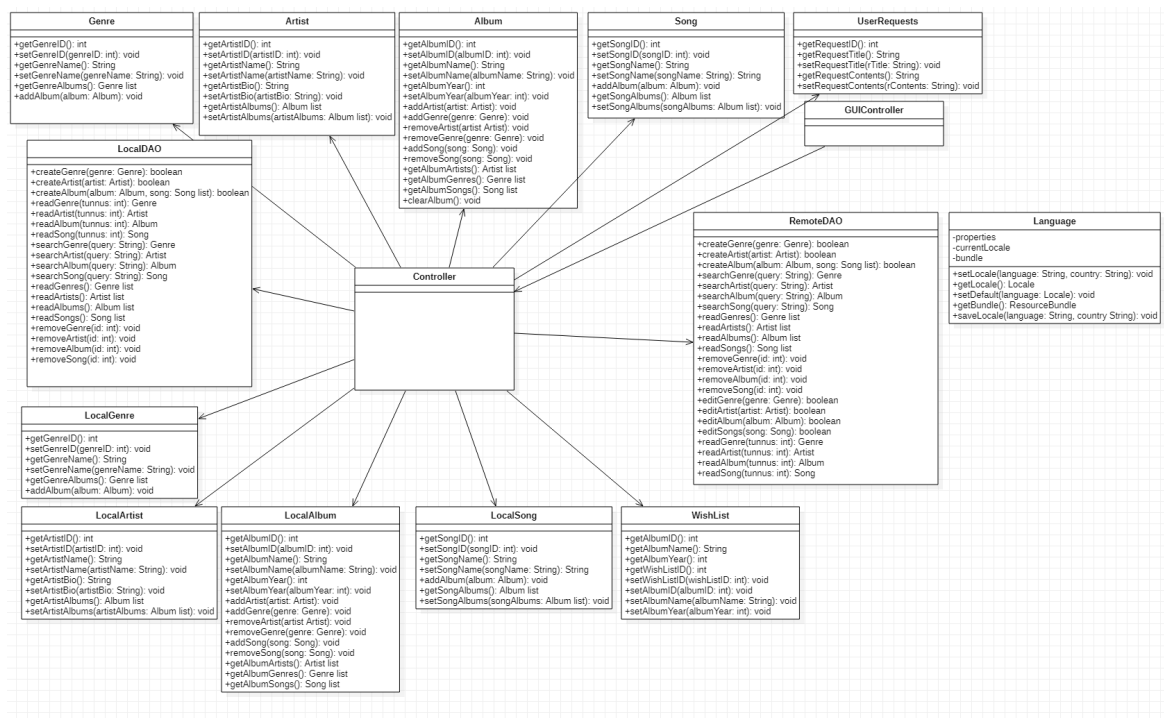
MusicArchin luokat on jaettu seuraaviin pakkauksiin.



Hibernate-pakkaus sisältää tietokantoihin liittyvät luokat, sekä

Hibernate-annotoidut olio-luokat, että etä- ja lähitietokantoihin yhteyttä ottavat luokat.

Album-, Artist-, Genre- Song- ja UserRequests-luokat käsittävät etätietokannan olioluokat.



Luokkakaavio keskeisimmistä luokista ja niiden metodeista.

| Controller |
|---|
| <pre> +createGenre(genreName: String): void +createArtist(artistName: String, artistBio: String): void +createAlbum(albumName: String, albumYear: int, genreListGiven: String table, artistListGiven: String table, songListGiven: String table): void +createLocalGenre(genreID: int, genreName: String): void +createLocalArtist(artistID: int, artistName: String): void +createLocalAlbum(albumName: String, albumYear: int, genreListGiven: String table, artistListGiven: String table, songListGiven: String table): void +getAlbum(albumID: int): void +getAlbumArtistList(albumID: int): Artist list +getAlbumGenreList(albumID: int): Genre list +getAlbumSongList(albumID: int): Song list +editGenre(genreID: int, genreName: String): void +editArtist(artistID: int, artistName: String): void +editAlbum(albumID: int, albumName: String, albumYear: int, artistListEdit: String table, genreListEdit: String table): void +editLocalAlbumDescription(localAlbum: LocalAlbum): void +getLocalAlbumDescription(id: int): String +removeGenre(genreID: int): void +removeArtist(artistID: int): void +removeAlbum(albumID: int): void +removeLocalGenre(genreID: int): void +removeLocalArtist(artistID: int): void +removeLocalAlbum(albumID: int): void +searchGenre(genreName: String): Genre +searchArtist(artistName: String): Artist +searchAlbum(albumName: String): Album +getGenres(): Genre table +getArtists(): Artist table +getAlbums(): Album table +getSongs(): Song table +getSearchable(): String list +getSearchableGenres(): String list +getSearchableArtists(): String list +getSearchableAlbums(): String list +getSearchableSongs(): String song +getGenreAlbums(genreID: int): Album list +getArtistAlbums(artistID: int): Album list +getAlbumSongs(albumID: int): Song list +addToWishList(albumID: int, albumName: String, albumYear: int): boolean +readWishList(): Wishlist list +removeFromWishList(id: int): boolean +searchWishList(albumID: int): boolean +createRequest(rTitle: String, rContents: String): void +getRequest(id: int): UserRequests +getRequests(): UserRequests table +removeRequest(id: int): boolean </pre> |

Lisäksi Controller-luokan luokkakaavio.

Jokainen näistä luokista on annotoitu Hibernate-API:n mukaisesti, joten tietueiden lisääminen, muokkaaminen ja poistaminen tietokannasta onnistuu helpoiten.

Lisäksi, Hibernate.Maven-pakkauksen sisällä on myös sekä RemoteDAO-, että LocalDAO-luokat, joilla otetaan yhteys etä- ja asiakkaan omaan lähitietokantaan. Näiden luokkien sisältö on laajalti sama, sillä molemmat koostuvat metodeista joilla otetaan yhteys tietokantaan, luodaan/haetaan/muokataan/poistetaan tietueet edellämainittujen Hibernate-luokkien avulla.

Controller-pakkaus koostuu luokista joilla hallinnoidaan sovelluksen eri osia. Esimerkiksi datan haku tietokannasta, sekä sen käsitleminen oliona sovelluksen sisällä hoidetaan tässä pakkauksessa. Myös käyttöliittymäkontrollerit löytyvät tästä pakkauksesta.

Controller-luokka on sovelluksen pääkontrolleri, joka käsittelee datan siirron sovelluksen, etätietokannan, sekä asiakkaan paikallisen tietokannan välillä. Sovelluksen muut osat käyttävät sitä luomaan uusia tietueita, sekä hakemaan tietoja molemmista tietokannoista.

AlbumPageController-luokan tarkoitus on hallinnoida albumi-näkymää. Sen on tarkoitus luoda näkymä dynaamisesti, riippuen siitä minkä albumin tietoja asiakas on käyttöliittymässä pyytänyt. Tämän kontrollerin konstruktori saa parametreikseen siis sekä Controllerin, että pyydetyn albumin ID:n, jonka avulla voidaan tietokannasta hakea albumin genrejä, artisteja, sekä kappaleita. Nämä asetetaan sitten käyttöliittymässä näkyviin, kuten myös ne painikkeet, joilla asiakas voi lisätä albumin omaan lokaaliin kokoelmaansa tai toivelistaansa. Luokka myös luo ylläpidolle edit-painikkeen, jota painamalla albumin tietoja voidaan muokata etätietokannassa. Edit-painiketta painaessa tämä muuttuu Save-painikkeeksi, ja käyttöliittymään luodaan myös Delete-painike, jota painamalla albumi ja sen kappaleet poistetaan tietokannasta.

FrontPageController-luokka käsittelee sovelluksen päänäkymän luonnin. Tämän luokan sisällä näkymään luodaan kaksi Tab-listaa, jotka sisältävät listan tietokannasta löytyvistä genreistä ja artisteista. Kontrolleri myös asettaa listaa rakentaessa niille EventHandlerin, joita klikattaessa näkymän oikealle puolelle rakentuu ruudukko-näkymä klikatun genren/artistin sisältämistä albumeista. Kun ruudukosta klikataan haluttua albumia, kontrolleri lähettää pyynnön AlbumPageControllerille, joka luo näkymän kyseiselle albumille lähettämällä sen ID:n eteenpäin. FrontPageController myös asettaa edit-painikkeen ylläpidolle näkyviin kun valitaan jokin artisti tai genre. Tällöin artistin/genren tekstikenttä muuttuu vapaasti muokattavaksi. Edit-painike muuttuu Save-painikkeeksi, jota painamalla muokkauskäsky lähetetään etätietokantaan. Myös Delete-painike luodaan, jota painamalla artisti/genre poistetaan. Molemmissa tapauksissa avataan näkyviin ponnahdusikkuna, joka vaatii

ylläpitoa varmistamaan muokkauksen ja poiston. Poiston yhteydessä ylläpitoa myös varoitetaan siitä, että tämä poistaa kaikki artistiin/genreen liittyvät albumit.

HelpController-luokka luo näkymän sovelluksen "help"-osalle, jonka tarkoitus on sisältää ohje sovelluksen käyttöä varten.

UserCollectionController luo näkymän asiakkaan omaa kokoelmaa varten.

Ensin luodaan lista lokaalista tietokannasta löytyvistä albumeista, joita klikattaessa näkymä tuo esille albumin tiedot. Genret, artistit sekä kappaleet. Asiakkaan on myös mahdollista tulevaisuudessa luoda kuvaus albumille vapaamuotoisella tekstikentällä, joka tallennetaan myös paikalliseen tietokantaan.

SearchController hallinnoi asiakkaan/ylläpidon suorittamaa hakua sovelluksen sisällä. Sille asetetaan parametreiksi sekä hakusana, että Controller-olio.

Luokka sitten hakee hakusanaa käyttäen kaikki artistit, genret sekä albumit tietokannasta, ja luo näkymän hakutuloksia varten. Tulokset esitetään GridPanea käyttäen. Mikäli haulla ei löytynyt mitään, tästä myös ilmoitetaan käyttäjälle. Käyttöliittymään myös tässä tapauksessa luodaan painike, jolla avataan lisäyspyyntölomake. Asiakas voi tähän vapaasti kirjoittaa haluamansa tietueen tiedon, joka sitten lähetetään etätietokantaan ylläpidon luettavaksi.

RequestFormsController- luokka luo näkymän pyyntölomakkeita varten. Tämä sisältää erinäiset tekstikentät eri tietueita varten. Genrejä lisättäessä nimi, artisteille asetetaan nimi, sekä lyhyt vapaamuotoinen biografia, albumeille genret, artistit sekä kappaleet, että albumin nimi ja julkaisuvuosi. Jokaisen tiedon lisäämiselle luodaan myös painike, joka lähettää tiedot Controllerin avulla etätietokantaan kaikkien saataville.

GUIController on käyttöliittymän pääkontrolleri. Tällä hallinnoidaan eri näkymien esittäminen asiakkaalle. Käyttöliittymä sisältää eri painikkeita, joita painamalla päästään sovelluksen eri osiin. GUIController myös sisältää metodin, jolla haetaan tietokannasta lista kaikista tietueista, mitä sieltä löytyy. Luokka käyttää

tätä listaa luomaan asiakkaalle ennustavan tekstin hakukentän sisällä, luomalla ContextMenuun eri listakohtia, jotka sisältävät asiakashaun kanssa yhteensopivan tekstin.

Model-pakkaus sisältää paikallisen tietokannan käyttämien olioiden luokat.

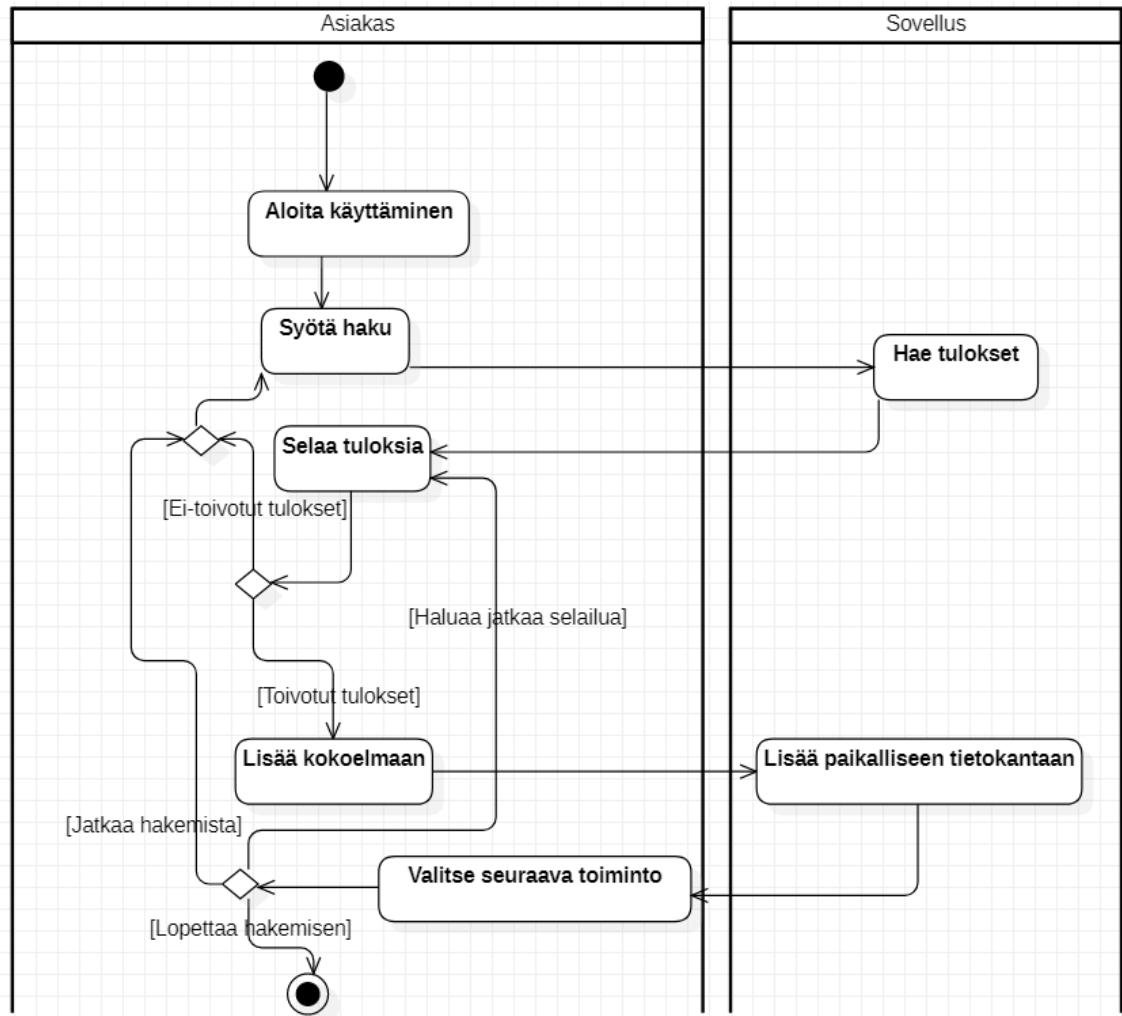
LocalAlbum, LocalArtist, LocalGenre ja LocalSong-luokat ovat samanlaisia Hibernate-annotoituja luokkia, kuten edellämainitut. Ainoa ero näiden välillä on se, että paikallista tietokantaa käyttävät oliot eivät automaattisesti luo omia tunnuksiaan, vaan ne saavat kyseiset ID:t etätietokannasta silloin, kun ne tallennetaan paikalliseen.

7 Ohjelmiston toiminta

MusicArchin toiminta perustuu ylläpidon tietueiden lisäämiseen, sekä asiakkaan paikalliseen tietokantaan tallentamiseen.

Kun tietoa halutaan tallentaa etätietokantaan, ylläpito lisää halutut tietueet käyttäen sovelluksen lisäyspyyntö-lomaketta. Tämä lomake ei anna ylläpidon lisätä tyhjiä kohtia, eikä sellaisia tietoja, jotka nojaavat aiempaan tietoon jota ei löydy tietokannasta. Käytännössä, mikäli ylläpito on lisäämässä albumia, jonka genreä tai artistia ei löydy tietokannasta, sovellus pakottaa lisäämään ne ensin. Muutoin välitauluihin tallennus ei onnistu oikein.

Lomake lähettää saadut tiedot eteenpäin kontrollerille List<String>-muodossa. Tämä puolestaan luo halutut oliot (Genre, Artist, Album, Song tai UserRequests lisätyn tietueen perusteella) ja lähettää ne RemoteDAO-luokalle käsiteltäväksi. Täällä sovellus tallentaa ne etätietokannan tietovarastoon, josta asiakas sekä ylläpito voivat tarkastella tietoja haluamallaan tavalla.



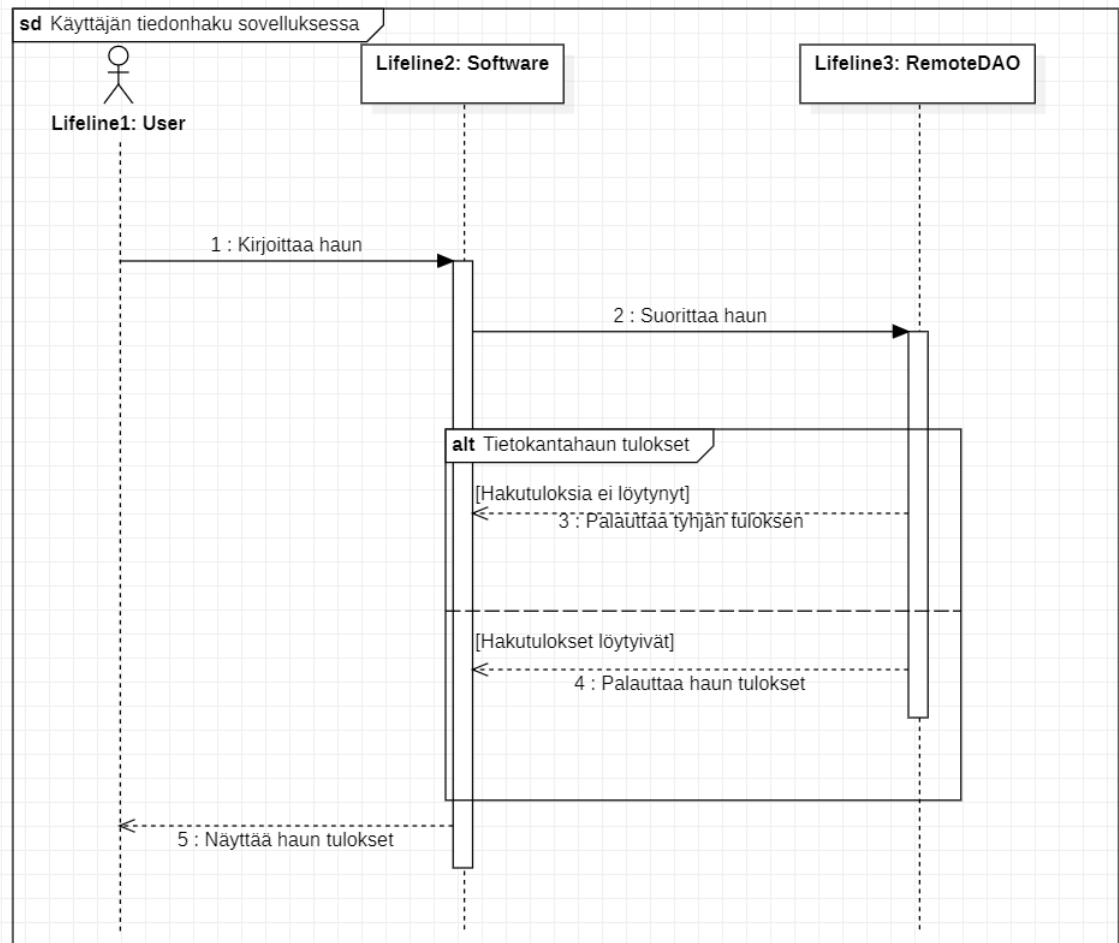
Aktiviteettikaavio asiakkaan albumihausta, sekä kokoelmaan lisäämisestä.

Kun asiakas haluaa tallentaa tietyn albumin kokoelmaansa, hänen täytyy ensin etsiä se tietokannasta. Tällöin käytetään sovelluksen hakutoimintaa.

Hakutoiminto on toteutettu niin, että hakupalkkia klikattaessa sovellus hankkii listan kaikista nimistä, jotka löytyvät tietokannasta. Tämä sisältää siis eri genrejen, artistien, albumien sekä kappaleiden nimet. Listasta karsitaan turha toisto, ettei asiakasta kuormiteta sillä liikaa. Esimerkiksi Black Sabbath, joka on sekä albumi, artisti ja kappale, löytyy tästä hakuehdotuslistasta vain kerran kolmen sijaan. Joka kerta kun asiakas lisää kirjaimen hakupalkkiin, luodaan ContextMenuun lista kaikista niistä nimistä, jotka vastaavat asiakkaan siihen asti kirjoittamaa tekstiä. Näin autetaan asiakasta löytämään haluttu tietue.

Mikäli asiakkaan hakemaa tietuetta ei löytynyt, sovellus ensin esittää haun tulokset. Tässä tapauksessa haun tulokset ovat, ettei haulla löytynyt mistään tietueesta mitään asiakkaan hakemaan. Sovellus sitten antaa hänelle mahdollisuuden lähettää lisäyspyyntö, jonka lomakkeen voi aktivoida painamalla käyttöliittymään esiintyvää painiketta. Tällöin käyttöliittymään tulee lomake jolla kaksi tekstikenttää, otsikko ja sisältö. Asiakas voi täyttää nämä haluamallaan tavoin, esimerkiksi, jos asiakas haluaa albumin, jonka genreä ei löydy tietokannasta, sovellus ei estä pyynnön lähettämistä. Itse pyyntö tallennetaan myös etätietokantaan, josta ylläpito voi sitä tarkastella.

Halutun albumin löydyttyä, asiakas voi napin painalluksella lisätä sen omaan kokoelmaansa. Tällöin sovellus ottaa käsiteltäväkseen albumin ID:n, jonka se lähettää kontrollerin kautta RemoteDAO:lle. Täällä ohjelma hakee kaiken albumiin liittyvän tiedon (Albumin nimen, julkaisuvuoden, albumin kappaleet, artistit sekä genret) ja tallentaa ne LocalDAO:a käyttäen asiakkaan omaan paikalliseen tietokantaan. Tällöin asiakas ei ole riippuvainen yhteydestä etätietokantaan, kun hän haluaa vain tarkastella omaa kokoelmaansa.



Sekvenssikaavio, joka esittää miten sovellus käsittelee asiakkaan hakupyynnön.

8 Kehitysprosessi ja kehitysvaiheen tekniikat

Ohjelmistotuotantoprojekti 1

Käytimme sovelluksen kehitykseen **Scrum**-menetelmää. Projekti oli jaettu kahteen erilliseen osaan, joista ensimmäinen (OTP1) oli jaettu neljään kahden viikon sprinttiin.

Aluksi ideoimme projektin visiota sovelluksen rajoitusten ja vaatimusten pohjalta. Päädyimme pian eräänlaiseen musiikkikirjastoon, jonka avulla asiakas voisi tallentaa mieleisensä (esimerkiksi fyysisesti omistamansa) albumit omaan albumikokoelmaan. Hän pystyisi myös etsimään tietoa genreistä, albumeista ja

artisteista. Alunperin ajattelimme sovellusta tietopankkina, mutta sprinttien edetessä siitä oli enemmän kokoelmapainotteinen.

Projektinhallintaan käytimme Nektionia ja projektin automatisoituun koontiin **Maven**-koontityökalua. Versionhallintaan käytimme **Git**-järjestelmää ja **GitLab**-verkkopalvelua. Tilasimme Metropolian **educcloud**-pilvestä palvelinkoneen tietokantaamme varten. Käytimme koodaukseen **Eclipse**-ohjelmointiympäristöä.

Suunnittelimme ja loimme käyttäjätarinat, joiden pohjalta suunnittelimme sovelluksen toimintaa. Määrittelimme käyttäjäpolun, jossa kävimme läpi asiakkaan kosketuspisteet sovellukseen.

Tämän jälkeen suunnittelimme sovelluksen tietokannan ja käyttäjäkeskeisen käyttöliittymän prototyypin. Suoritimme projektin pohjaksi datamallinnuksen, mihin kuului tietokannan rakennetta esittävä ER-kaavio, sekä tämän pohjalta generoitu relaatiotietokanta. Käytimme ER-kaavion ja relaatiotietokannan tekoon **ERDPlus**-työkalua.

Teimme sen jälkeen **StarUML**-mallintamisohjelmistolla ohjelmiston ulkoista vuorovaikutusta kuvaavan käyttötapauskaavion ja ohjelman keskeistä (Asiakas etsii albumia sovelluksen hausta ja lisää sen kokoelmaansa) käyttötapausta kuvaavan aktiviteettikaavion.

Olio-relaatiomuunnoksiin käytimme **Hibernate**-kirjastoa ja käyttöliittymän prototyypin suunnitteluun käytimme **InVision**-suunnittelutyökalua.

Suunnittelun jälkeen alustimme sovelluksen koodipohjan ja aloimme toteuttamaan näkymiä **Scene Builder**illa. Ohjelman rakenteen kuvaamista varten loimme luokkakaavion ja ohjelmiston viestinvälitystä (asiakkaan hakupyynnön käsittely) kuvaavan sekvenssikaavion. Toteutimme sovellusta **MVC**-mallin mukaisesti (Model, View & Controller). Modularisoimme sovellusta

aiemman luokkakaavion pohjalta siten, että päivitimme luokkakaavion vastaamaan sovelluksen sen hetkistä näkemystä ja laadimme luokista tarvittavat pakkaukset (DAO, Controller, Hibernate ja Model). Suunnittelimme myös rajapinnat luokille, joista myöhemmin voisi tulla vaihtoehtoisia toteutuksia (Language-rajapinta sovelluksen kielen vaihtoa varten ja Serializable Hibernate-luokkien sarjallistamiseen). Kävimme läpi sovelluksen käyttöliittymän virheiden ennaltaehkäisyn, niistä toipumisen sekä ”anteeksiantavuuden” näkökulmasta. Laadimme sovellusta varten myös sijoittelukaavion.

Ohjelmistotuotantoprojekti 2

Toinen osa (OTP2) oli jaettu kahteen kolmen viikon mittaiseen, sekä yhteen noin kahden viikon mittaiseen sprinttiin. Tänä aikana ryhmän keskeisin määränpää oli lokalisoida sovellus, tuoda siihen lisää tarvittavia ominaisuuksia ja parantaa sen toimivuutta.

Sprintin 5 aikana implementoimme koodin ominaisuuksia käyttöliittymää, ja alustettua sovellusta lokalisaatiota varten.

Sprintin 6 aikana aloitimme lokalisaation ja päivitimme käyttöliittymän kovakoodattuja tekstejä avaimiksi siten, että ne voitaisiin vaihtaa suomeksi tai englanniksi yhdellä painikkeella. Paransimme myös virheiden hallintaa antamalla virheistä ilmoituksia ja toimme uusia ominaisuuksia sovellukseen.

Sprintin 7 aikana saimme lokalisaation valmiiksi. Kaikki kovakoodaukset muutettiin avaimiksi siten, että sovelluksen kielen saa helposti vaihdettua. Saimme valmiiksi myös muokkaus ja poisto-ominaisuuksia, sekä toivelista ominaisuuden. Saimme parannettua myös haku-ominaisuutta siten, että haun avulla pääsee suoraan albumin sivulle. Myös lisäyspyyntö-ominaisuus saatiin tuotua sovellukseen kunnolla.

Projektin JAR-tiedoston luomiseen käytimme **Mavenin Shade & Dependency** liitännäisiä. Näiden liitännäisten tarkoitus oli tuoda kaikki riippuvuudet mukaan JAR-tiedostoon ja mahdollistaa tiedoston toimivuus.

Testaus

Testasimme jatkuvasti kehityksen aikana eri tietueiden lisäystä erilaisilla syötteillä, esimerkiksi siten, että albumia tehdessä ei anneta ylläpidon syöttää kirjaimia albumin julkaisuvuosi-kenttään, tai jättää mitään muita kenttiä tyhjäksi.

Käytimme automaattisiin yksikkötesteihin JUnit-kirjastoa. Teimme automaattiset testit etätietokantaan tallentamista varten, joilla varmistamme sen, että tietokantaan voi tallentaa vain aidosti validia dataa. Esimerkiksi testataan sitä, että albumin nimeä ei voi jättää tyhjäksi. Testaamme myös sitä, että Genrejä, Artisteja sekä Albumeita voidaan sekä lisätä, etsiä, että poistaa etätietokannasta.

| name | instruction | branch | complexity | line | method | class |
|-----------|----------------------|--------------------|--------------------|----------------------|--------------------|-------------------|
| Album | M: 90 C: 32 26% | M: 6 C: 0 0% | M: 14 C: 5 26% | M: 28 C: 10 26% | M: 11 C: 5 31% | M: 0 C: 1 100% |
| Artist | M: 19 C: 26 58% | M: 1 C: 1 50% | M: 5 C: 6 55% | M: 7 C: 10 59% | M: 4 C: 6 60% | M: 0 C: 1 100% |
| Genre | M: 16 C: 22 58% | M: 1 C: 1 50% | M: 4 C: 5 56% | M: 6 C: 8 57% | M: 3 C: 5 63% | M: 0 C: 1 100% |
| LocalDAO | M: 1063 C: 17 2% | M: 69 C: 1 1% | M: 60 C: 2 3% | M: 341 C: 8 2% | M: 25 C: 2 7% | M: 0 C: 1 100% |
| RemoteDAO | M: 741 C: 656 47% | M: 76 C: 24 24% | M: 61 C: 27 31% | M: 270 C: 181 40% | M: 18 C: 20 53% | M: 0 C: 1 100% |
| Song | M: 16 C: 22 58% | M: 1 C: 1 50% | M: 4 C: 5 56% | M: 6 C: 8 57% | M: 3 C: 5 63% | M: 0 C: 1 100% |

Etätietokannan hibernate-olioiden testikattavuus.

9 Jatkokehitysmahdollisuudet

Saavutimme Ohjelmistotuotantoprojekti 2 -kurssin aikana suurimman osan tavoitteistamme sovelluksen suhteen. Jatkuva kehitystyö on kuitenkin tärkeää sillä vain näin voidaan tuottaa aikansa mukaisia tuotteita ja palveluita. Olemme MusicArchin suhteen samaa mieltä jatkuvasta kehittämisestä ja siksi olemme ottaneet tämän huomioon suunnitellessamme ja toteuttaessamme tuotetta:

Sovellus ja sen tietokannat on suunniteltu joustavaksi ja sitä voidaan soveltaa erilaisten kokoelmien hallintaan. Tämä ominaisuus mahdollistaisi sovelluksen lisensoinnin erilaisille yrityksille ja organisaatioille.

Sovellukseen voitaisiin jatkossa tuoda mahdollisuus lisätä genrejä ja artisteja olemassa oleville albumeille.

Navigointia sovelluksessa voitaisiin helpottaa lisäämällä sivun polun näkyminen ja albumisivulla voitaisiin totetuttaa suuren datamäärän käsittelemiseksi albumien rajaaminen tiettyyn kappalemäärään näkymässä.

Käyttöliittymästä olisi hyvä luoda erilaisia variaatioita tai käyttöliittymän muokattavuutta voitaisiin kehittää. Esimerkiksi teeman vaihto, fonttikoon valinta tai ääniselitteet lisäisivät käyttömukavuutta ja mahdollistaisivat suuremmalle ihmisryhmälle mahdollisuuden käyttää sovellusta.

Sovellukseen kielivaihtoehtoja olisi hyvä lisätä, jotta mahdollisimman moni voisi saada käyttää tuotetta omalla äidinkielellään.

10 Yhteenveto

Dokumentissa käsitellään sovelluksen käyttöohjeet, sekä tekninen dokumentaatio. Jälkimmäinen sisältää tuotteen vaatimukset, käyttäjäroolit- ja tapaukset, projektin käsitteet, määritelmät, sekä ohjelmiston tietomalli. Lisäksi käymme läpi ohjelmiston rakennetta ja toimintaa, kehitysprosessia ja sen tekniikoita, sekä jatkokehitysmahdollisuuksia. Aiheille ollaan liitetty niihin liittyvät kaaviot.

Projektissa saavutimme suurimman osan siitä, mitä sen visiossa haettiin. Projektin toisen osan aikana mahdollistettiin sovelluksen kielenvaihto ja sovellusta kehitettiin muutenkin merkittävästi eteenpäin.

Scrum teki ryhmän työskentelystä järjestelmällisestä ja ryhmä tiesi selkeästi mitä milloinkin oli työn alla.