Antti Aho, Andreas Lang, Jaakko Kuivasniemi, Jesper Oja

LFG OHTU2

Looking For Group

Metropolia
Ammattikorkeakoulu Insinööri
(AMK)

Tieto- ja viestintäteknologian tutkinto-ohjelma

1 Johdanto

LFG on sosiaalinen verkosto nimenomaan pelaajille, jossa käyttäjät voivat lisätä profiiliinsa pelejä, katsoa mitä muut pelaavat, tehdä postauksia, jakaa pelejään sekä kommentoida ja tykätä tai “dislikettää” postauksista.

Dokumentista saa hyvän kuvauksen koko LFG:n toiminnasta ohjelmana.

Projektin Backend on rakennettu Microsoftin ASP.NET frameworkillä käyttäen MySQL tietokantaa, ja Front-end VUE Frameworkkiä hyödyntäen TypeScriptiä.

2 Tuotteen vaatimukset

Tärkeimmät tuotetavoitteet: Olla pelaajan apuväline ja sosiaalinen alusta, johon voi luoda profiilin ja tähän merkitä pelejä, joita käyttäjä pelaa. Alustalla pelaajat voivat tutustua ja vertailla sijoituksiaan (rankkeja), muodostaa tiimejä sekä etsiä peliseuraa ja jakaa pelejään.

Tuote on kohdistettu kaikenikäisille ja tasoisille pelaajille, eSport-tiimeille sekä turnauksien järjestäjille ym.

Käyttäjä pystyy ottamaan yhteyttä toisiin pelaajiin, sekä alusta helpottaa verkostoitumista peliyhteisöissä. Tätä toiminnallisuutta ei tosin saatu valmiiksi.

Juuri vastaavanlaista palvelua ei oikeastaan vielä ole, joten tuote on uniikki. Lähimmät tuotteen kaltaiset ovat Discord, Steam community sekä LinkedIn.

Alustalla käyttäjä voi:

- Luoda profiilin ja ilmoittaa siellä mitä kaikkea ja millä tasolla käyttäjä pelaa.

- Voi tehdä postauksia, nähdä muiden postauksia, sekä kommentoida niitä.
- Alustalla kirjaudutaan Discord käyttäjällä
- Julkaista päivityksiä.

Näillä toiminnoilla pelaaja voi helposti löytää itselleen peliseuraa, löytää kavereita sekä jakaa onnistumisiaan peleissä.

3 Käyttäjäroolit ja käyttötapaukset

Käyttäjärooleja on siis vain yksi, normaali käyttäjä.

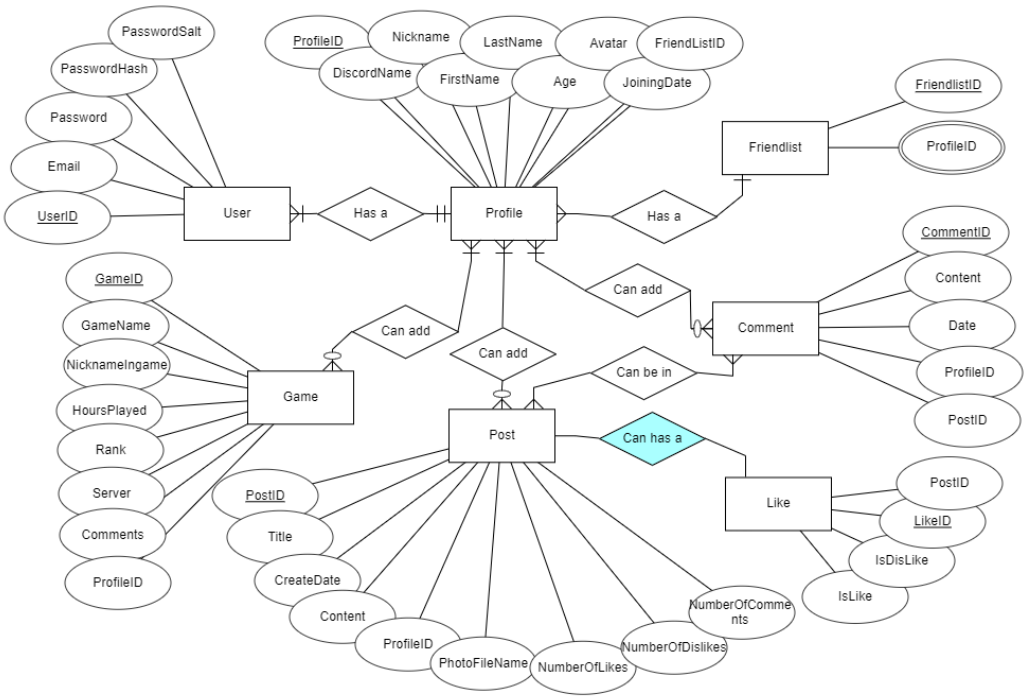
Käyttäjä kirjautuu palveluun käyttäen Discord APIa, joka hakee profiilitiedot käyttäjän profiiliin.

Kun käyttäjä on kirjautunut sisään, hän voi lisätä pelejä omaan profiilinsa, tehdä postauksia, ja kommentoida pelejä ja postauksia. Näitä kaikkia voi myös jälkikäteen muokata. Postauksilla on myös alapeukku (Like) ja yläpeukku (Dislike) nappulat.

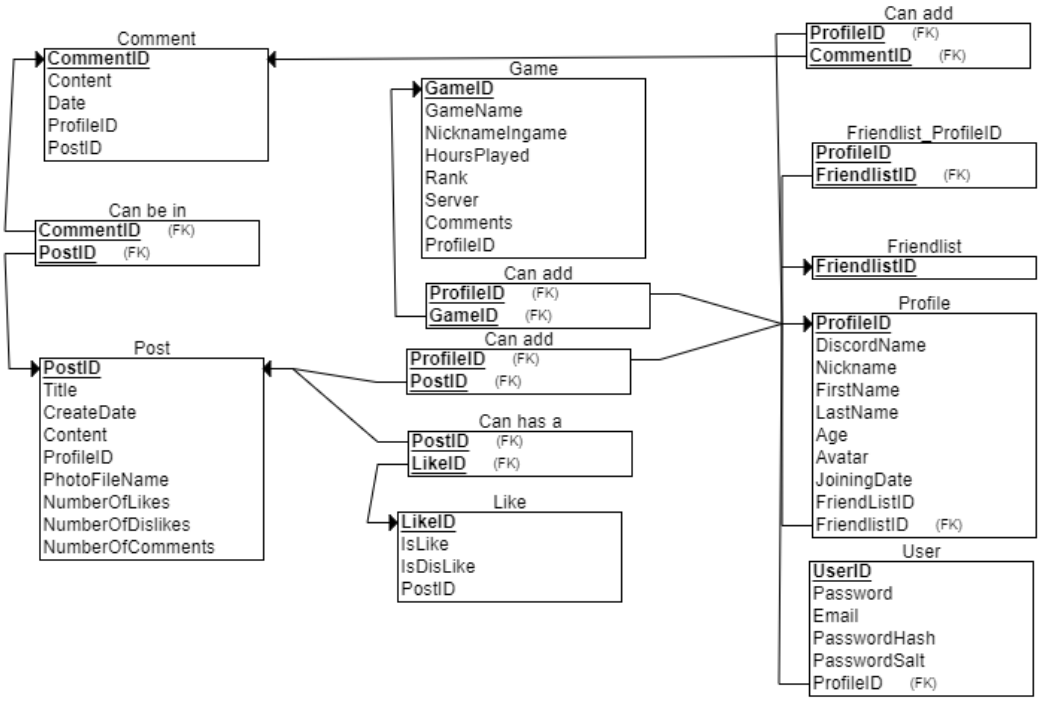
Kuva 1

4 Ohjelmiston tietomalli

Entity-Relation kaavio projektimme päivitetystä tietokannasta (Kuva 2), ja sen alta löytyy ER kaaviosta johdettu projektin tietokannan Relaatiokaavio (Kuva 3).



Kuva 2



Kuva 3

5 Ohjelmiston rakenne

Kuvassa 3 LFG:n luokkakaavio. Projekti toteutettiin ASP.Netillä ja VUE:lla, joten varsinaisia pakkauksia ei ole kuten Javassa, mutta Model, Controller ja View namespacet löytyy koska projekti toteutettiin MVC mallin mukaan, tosin View:iä toteuttaa Vue, dataa taas namespace Model ja toiminnallisuutta sekä näiden välistä liikennettä namespace Controller. React on vaihdettu Vueen OTP2:ssa.

6 Kehitysprosessi ja kehitysvaiheen tekniikat

Front-end

- Vue käyttäen Typescript ja tukevia kirjastoja:
- Käytimme tailwindcss tyylikirjastona
- Google Icons useaan ikoniin.
- Animate CSS muutamiin css animaatioihin.
- Discord API kirjautumiseen

Back-end

- ASP.NET käyttäen .NET Core 3.1 kirjastoa C#-ohjelmointikielen kanssa.

Tietokanta

- Tietokanta on tehty MySQL-tietokanta mysql.metropolia.fi-palvelimella, johon on luotu yhden tiimin jäsenen omalla Metropolian käyttäjätunnuksella.

Testaus

Testauksessa käytimme Vue-test-utils ja Vitest komponentti testaamiseen, sekä Playwright end-to-end testaamiseen. Jatkuvaa kehitystä (CI) käytimme alussa Jenkkinsiä, mutta meillä oli suuria ongelmia sen kanssa. Opettajan myöntymyksellä, vaihdoimme CI osuuden Azure devOpsiin, jossa saatiin todella nopeasti ja helposti suoritettua sekä frontendin, että backendin buildaus, mikä oli Jenkkinsillä todella vaikeaa. Valitettavasti emme ole saaneet kummassakaan CI ympäristössä testausta toimimaan.

7 Jatkokehitysideoita

Yhteisöt jokaiselle pelille

- meillä ei ollut tarpeeksi aikaa kehittää tätä ominaisuutta, mutta alkuperäisessä suunnitelmassa tämä ominaisuus olisi otettu käyttöön tällä ajanjaksolla.

Parempi Games API

- tietojen saamista nopeammin ja laajemmin mm. esimerkkejä peleistä (vaatii ehkä maksullisen API-version)

Ystävien kutsu

- on parempi pitää yhteyttä omiin ystäviin

Käyttäjälle räätälöity syöte

- vain ystävien syötteet näytetään, ei kaikkien syötteitä

Parempi automaattinen testaus

- Käyttää Azurea entistä paremmin kehityksen tukena, ajamalla säännöllisesti testejä siellä.

8 Yhteenveto

Tavoitteemme ohjelmistotuotantoprojekti kakkosessa oli saada jatkettua ensimmäistä projektia toteuttamalla siihen lokalisaation, ottaa käyttöön jokin peli API, luoda Community-ominaisuus peleille, luoda kaverilistaus ja käyttäjille mahdollisuus lisätä toisia käyttäjiä kaverilistalle, sekä vaihtaa frontend kehitysympäristö Reactista Vuehen.

Onnistuimme tavoitteissa muuten, paitsi kaverilistan toteutus jäi tekemättä. Vaihdoimme kirjautumisen Discord login, mikä ei ollut suunnitelmissa alunperin, mutta mikä on tietoturvallisuuden kannalta todella suuri asia.