Risbeegomfkerho

Toteutusdokumentti

Sisällysluettelo

Sisällys

[2 Yleiskuva 3](#_Toc88647158)

[3 Asennus 3](#_Toc88647159)

[3.1 Ympäristömuuttujat 3](#_Toc88647160)

[4 Muutokset 4](#_Toc88647161)

[5 Komponentit 4](#_Toc88647162)

[5.1 Backend 4](#_Toc88647163)

[5.2 Frontend 5](#_Toc88647164)

# Yleiskuva

Risbeegomfkerho on Vantaalla sijaitseva frisbeegolfia pelaava kerho. Kerholla on jäseniä noin kymmenen. Kerho järjestää aika-ajoin kerhokilpailuja.

Suurin osa kerholaisista käyttää tulosten ottamiseen maksullista UDisc sovellusta. Haittapuolena sovelluksessa on se, että se ei mahdollista tasoitusten laskemista, jolloin heikommat kerholaiset eivät koskaan pääse nauttimaan voiton tuomasta maineesta ja kunniasta.

Sivuston idea on siirtää tulokset UDiscistä saatavan csv-tiedoston avulla järjestelmään, laskea jokaiselle pelaajalle ratakohtainen tasoitus ja näyttää jokaisen kilpailun yhteydessä tasoitettu tulos. Lisäksi jokaisesta kilpailusta jaetaan pisteitä, jotka ovat riippuvaisia pelaajan tasoitetusta sijoituksesta.

Sivuston laskee kilpaluksi esimerkiksi kaikki kierrokset, joissa on ollut mukana vähintään viisi kerholaista ja jotka osuvat tietyn aikavälin sisään. Ylläpitäjä voi muuttaa em. rajoja.

Järjestelmä on testattu yleisimmillä selaimilla (Chrome, Firefox, Edge).

# Asennus

Vaatii NodeJS. Tehty käyttämään versiota 14.16.0 tai uudempaa.

Asenna riippuvuudet sovelluksen juurihakemistossa suorittamalla komento

**npm install**

Käännä backendin TypeSript -tiedostot JavaScriptiksi komennolla. Backend kääntyy hakemistoon /server

**npm run build-backend**

Käännä frontendin lähdekoodi html-sivuksi/javascriptiksi komennolla. Frontend kääntyy hakemistoon /client

**npm run build-frontend**

Juurihakemistoon tulisi ilmestyä kaksi uutta kansiota; server ja client. Palvelimen voi käynnistää komennolla

**npm start**

Palvelin käynnistyy oletusarvoisesti porttiin *8080* (http://localhost:8080). Porttia voi muuttaa asettamalla ympäristömuuttuja *PORT*.

## Ympäristömuuttujat

DB\_HOST = <Tietokannan osoite>

DB\_USER = <Tietokannan käyttäjä>

DB\_PASS = <Tietokannan käyttäjän salasana>

TOKEN\_KEY = <Salasana, jolla salataan sisäänkirjautumisen tokenit>

Ympäristömuuttujat on mahdollista asentaa luomalla juurihakemistoon .env-tiedoston ja lisäämällä em. muuttujat sinne.

# Muutokset

## Tietokanta

*Index*in ollessa varattu sana MySQL:ssä, vaihdettiin taulujen Score ja Hole index-attribuutti muotoon **indeksi**

Lisätty taulukko *logs* lokitietoja varten

*Scorecard* taulu yhdistetään käyttäjään yhdistämällä playerName *alias*-taulun kautta *user*-tauluun sen sijaan että jokaiselle Scorecard riville haettaisiin *user*-avain.

*Hole*-taulua todettiin tarpeettomaksi ja sitä ei otettu käyttöön.

# Palvelin

## Toteutus

Palvelin toteutettiin Javascriptillä (NodeJS) Typescript syntaksia käyttäen. Koodin tyylittelyn noudattaa eslintin perusssääntöjä.

## Rakenne

**/graphql** - Sisältää GraphQl mutaatiot, kyselyt ja skeemat.

**/services** - Palvelut taustan ja tietokannan välillä

|  |  |
| --- | --- |
| Komponentti | Kuvaus |
| gameServices | Yhteys tietokannan ja pelin osien (tuloskortti, rata, yms.) välillä |
| logServices | Hakee ja kirjoittaa lokitietoja tietokantaan |
| setupService | Yhteys tietokannan kilpailuasetuksiin |
| statsService | Noutaa tilastotietoja tietokannasta |
| userService | Hoitaa käyttäjien ja heidän aliaksien lisäämisen, muokkaamisen ja poistamisen tietokantaan/tietokannasta |

**/upload** - Kansioon tallennetaan käyttäjän lähettämä .csv -tiedosto muodossa <tunnu>.csv

**/utils** -Sisältää apufunktioita

|  |  |
| --- | --- |
| Komponentti | Kuvaus |
| csvParser | Parsii .csv tiedoston JSON muotoon sekä kirjoittaa sen tietokantaan |
| mathAndShit | Matemaattisia funktioita |
| mySqlHelpers | Apufunktioita tietokannan käsittelyyn. |
| validators | TypeScriptin string- ja int-validaattorit |

# Frontend

## Toteutus

Frontend toteutettiin Javascriptiä ja REACT-kirjastoa käyttäen. Osia ulkoasuun saatiin Material-Ui kirjastosta. Lähdekoodin noudattaa Eslintin airbnb tyylisääntöjä.

## Rakenne

**/assets** - Sisältää sivustolla käytettävät kuvatiedostot

**/components** - Komponentteja

|  |  |
| --- | --- |
| Komponentti | Kuvaus |
| Aliases | Käyttäjän aliaksien näyttäminen ja hallinnointi |
| AppBar | Sivuston yläreunassa oleva navipalkki |
| CreateAccount | Lomake ja tapahtumakäsittelijät tunnuksen luomiseen |
| FormikField | Lomakkeen tekstikenttä |
| HideIfNotLogged | Renderöi komponentin sisällä olevat komponentit vain jos käyttäjä on kirjautunut sisään tai käyttäjän rooli on komponentin argumentiksi annettu rooli. |
| LoadingPage | Lataussivu. Himmentää sivun ja näyttää pyörivän rinkulan |
| LoggedUserProvider | Toimittaa lapsikomponenteille tiedon onko käyttäjä kirjautunut sisään |
| LoginForm | Kirjautumislomake sekä siihen liittyvät tapahtumakäsittelijät |
| LogoutButton | Uloskirjautumisnappi |
| NaviLinkit | Sisältää valikoissa olevat navigointilinkit |
| UnActivatedUsers | Näyttää aktivoimattomat käyttäjä ja mahdollistaa niiden aktivoinnin & poistamisen |
| UploadCsv | .csv -tiedoston lähetys |

**/graphl** - GraphQl kyselyt ja mutaatiot

**/hooks** - Kustomoituja koukkuja

|  |  |
| --- | --- |
| Komponentti | Kuvaus |
| useAliases | Hakee, tallentaa ja poistaa aliaksia palvelimelta |
| useGetCompetitions | Hakee kilpailut palvelimelta ja parsii ne helposti käytettävään muotoon |
| useMe | Hakee kirjautuneen käyttäjän tiedot palvelimelta |
| useSetup | Hakee kilpailujen asetukset palvelimelta |

**/pages** - Sisältää sivut sekä mahdolliset niihin liittyvät komponentit

|  |  |
| --- | --- |
| Alihakemisto | Kuvaus |
| Admin | Käyttäjähallinnan sivu |
| Competitions | Kilpailut |
| Frontpage | Etusivu |
| Logs | Lokit |
| Settings | Asetukset |
| Stats | Stats |

**/utils** - Apufunktiot

|  |  |
| --- | --- |
| Tiedosto | Sisältö |
| apolloClient | Luo uuden apolloclientin ja määrittelee backendin osoitteen |
| competitionStuff | Laskee sijoitukset kilpailussa |
| dataAndTime | Muuntaa UTC-ajan paikalliseksi ajaksi |
| theme | Sivuton Material UI -teema |

# Yhteenveto

Projekti sujui ilman suurempia ongelmia. Eniten haasteita toi MySql kyselyjen luonti. Tavoite oli, että kaikki tarvittava data haetaan yhdellä kyselyllä.

Tietokantapalveluksi valittiin Amazonin RDS. Samalla päätös syntyi myös koko sovelluksen laittamisesta Amazoniin. Palvelu pyörii (ainakin kirjoitushetkellä) ongelmitta Amazonissa (<https://tinyurl.com/risbeegomfkerho>).

Isoin ongelma AWS:n kanssa oli saada järjestelmä kulkemaan salatun yhteyden yli. AWS tarjoaa ilmaisia sertifikaatteja, mutta ilmeisesti niiden käyttäminen vaatii oman domainin. Oman domainin puuttuessa luotiin itse allekirjoitettu sertifikaatti, joka saa selaimen näkemään punaista. Joka tapauksessa kaikki yhteydet taustaan kulkevat https -yhteyden kautta.