Tietokantasovelluksen dokumentaatio

1. Johdanto:

Järjestelmässä on kyse leaderboardista eli eräänlaisesta top-listasta. Käyttäjät voivat lisätä omia leaderboardeja ja omia suorituksiaan, joista saavat pisteitä, jotka sitten tulevat leaderboardiin ja niistä muodostuu käyttäjien rankkaus. Järjestelmä mahdollistaa siis omien leaderboardien lisäämisen ja niissä kilpailun.

Järjestelmä toteutaan Javalla käyttäen apuna Springiä. Sovelluksen tietokanta on PostgreSQL, mutta sen muuttaminen on helppoa esimerkiksi tsql:ksi.

2. Yleiskuva järjestelmästä:

Katso kuva: käyttötapauskaavio.png

Jokamies:
Kuka tahansa kuka tulee katsomaan sivustoa.

Osallistuja:
Rekisteröitynyt käyttäjä, joka tekee erilaisia urheilusuorituksia.

Admin:
Pisteyttää erilaiset suoritukset sekä hallinnoi muutenkin järjestelmää

3. Käyttötapaukset:

Jokamiehen käyttötapaukset:

*Leaderboardin luku:
Kuka tahansa voi katsoa leaderboardia ja sen tilannetta.

*Rekisteröityminen:
Rekisteröityminen tapahtuu lomakkeella, jossa annetaan erilaisia tietoja.
Heti rekisteröitymisen jälkeen jokamiehestä tulee osallistuja, jolloin hän saa osallistujan käyttöoikeudet.

Osallistujan käyttötapaukset:

*Oman suorituksen lisääminen:
Osallistuja voi lisätä oman suorituksen järjestelmään täyttämällä lomakkeen johon hän asettaa suorituksen lajin ja keston.

*Omien suoritusten katsominen:
Osallistuja voi katsoa omia suorituksiaan.

*Leaderboardin lisääminen:
Voi lisätä leaderboardin ja antaa sille nimen

*Käyttäjien lisääminen leaderboardiin
Voi lisätä käyttäjiä leaderboardiin

*Sisäänkirjautuminen:
Osallistuja voi kirjautua sisään omilla tunnuksillaan

Adminin käyttötapaukset:

*Suoritusten pisteytys:
Admin voi pisteyttää eri suorituksia eri arvoisiksi.

*Suoritusten poisto
Admin voi poistaa käyttäjien suorituksia

*Leaderboardin poistaminen:
Admin voi nollata leaderboardin, jolloin leaderboardi poistuu, mutta suoritukset jäävät jäljelle.

4. Järjestelmän tietosisältö

Katso kuva: tietosisältö.png

Tietokohteet:

Tietokohde: User

Attribuutti:	Arvojoukko:	Selitys:
id	Kokonaisluku	Identifioi jokaisen käyttäjän
username	Merkkijono	Käyttäjän käyttäjänimi
password	Merkkijono	Käyttäjän salasana
email	Merkkijono	Käyttäjän sähköpostiosoite
points	Kokonaisluku	Käyttäjän ansaitsemat pisteet, joita hän on saanut eri suorituksista
roleName	Merkkijono	Käyttäjän roolin nimi, esimerkiksi user tai admin

Tietokohde: Accomplishment

Attribuutti:	Arvojoukko:	Selitys:
Accomplishment_id	Kokonaisluku	Identifioi jokaisen suorituksen
sport	Merkkijono	Suorituksen laji
user_id	Kokonaisluku	Vierasavain User-tauluun, jokaisella suorituksella on yksi ja vain yksi käyttäjä
points	Kokonaisluku	Suorituksen pisteet
lengthinminutes	Kokonaisluku	Suorituksen pituus minuuteissa

Tietokohde: Leaderboard

Attribuutti:	Arvojoukko:	Selitys:
Leaderboard_id	Kokonaisluku	Identifioi jokaisen leaderboardin
name	Merkkijono	Leaderboardin nimi

Tietokohde: Leaderboard_users

Attribuutti:	Arvojoukko:	Selitys:
Leaderboard_users_id	Kokonaisluku	Identifioi jokaisen leaderboard - users parin
Leaderboard_id	Kokonaisluku	Vierasavain Leaderboard-tauluun
user_id	Kokonaisluku	Vierasavain User-tauluun

Linkkitaulu, joka liittää yhteen kaikki leaderboard - user parit.

5. Relaatiotietokantakaavio:

Katso kuva: Relaatietokantakaavio.png

```
CREATE TABLE ACCOMPLISHMENT
(ACCOMPLISHMENT_ID LONG NOT NULL GENERATED ALWAYS AS IDENTITY,
SPORT VARCHAR(255) NOT NULL,
USER_ID LONG NOT NULL REFERENCES USER(USER_ID),
POINTS INT(20),
LENGTHINMINUTES INT(20))

CREATE TABLE USER
(USER_ID LONG NOT NULL GENERATED ALWAYS AS IDENTITY,
USERNAME VARCHAR(255) NOT NULL,
PASSWORD VARCHAR(255) NOT NULL,
EMAIL VARCHAR(255) NOT NULL,
POINTS INT(20),
ROLE VARCHAR(255))

CREATE TABLE LEADERBOARD
(LEADERBOARD_ID LONG NOT NULL GENERATED ALWAYS AS IDENTITY,
NAME VARCHAR(255) NOT NULL)

CREATE TABLE LEADERBOARD_USERS
(LEADERBOARD_USERS_ID LONG NOT NULL GENERATED ALWAYS AS IDENTITY,
LEADERBOARD_ID LONG NOT NULL REFERENCES
LEADERBOARD(LEADERBOARD_ID),
USER_ID LONG NOT NULL REFERENCES USER(USER_ID))
```

6. järjestelmän yleisrakenne:

Tietokantasovellusta tehdessä on noudatettu MVC-mallia. Kontrollerit löytyvät liikuntaleaderboard.controller-packagesta, näkymät löytyvät jsp-kansiosta jsp-sivuina, mallit ovat liikuntaleaderboard.content-packagessa. Servicet löytyvät liikuntaleaderboard.service-packagesta ja ne tekevät suurimman osan työstä, SQLRepo-luokat ovat liikuntaleaderboard.repository-packagessa ja ne

tekevät kantakyselyt.

Sovellus on rakennettu käyttäen apuna Springiä. Projekti on maven projekti. Kyselyiden sanitoimiseksi on käytetty apuna Javan PreparedStatementtiä.

7. Järjestelmän komponentit:

accomplishments.jsp:
Suoritusten hallintasivu, lisää suoritus, muokkaa suoritusta tai poista suoritus

editLeaderboard.jsp:
Leaderboardien hallintasivu, lisää leaderboardi, lisää käyttäjiä leaderboardiin

leaderboard.jsp:
Yhden leaderboardin katseleminen, näyttää leaderboardin tilanteen

login.jsp:
Sisäänkirjautumissivu

mainpage.jsp:
Pääsivu, sisältää kaikki leaderboardit ja linkit leaderboardien sivuille

register.jsp:
Rekisteröitymissivu

userInfopage.jsp:
Käyttäjäntiedotsivu, näyttää tiedot käyttäjästä, käyttäjän suorituksen sekä käyttäjän leaderboardit

userpage.jsp:
Näyttää kaikki käyttäjät ja linkit käyttäjien sivuille

Content:
Accomplishment.java:
Suoritusluokka, sisältää getterit ja setterit suorituksen ominaisuuksille sekä lisäksi konstruktorit suoritukselle.
Leaderboard.java:
Leaderboardluokka, sisältää getterit ja setterit leaderboardin ominaisuuksille sekä lisäksi konstruktorit leaderboardille.
User.java:
Käyttäjäluokka, sisältää getterit ja setterit käyttäjän ominaisuuksille sekä lisäksi konstruktorit käyttäjälle sekä comparaattorin käyttäjille.

Controller:
AccomplishmentController.java:
Toteuttaa AccomplishmentControllerInterfacen määrittelemän rajapinnan. Kontrolleri kuuntelee suoritukseen liittyviä pyyntöjä ja ohjaa ne eteenpäin serviceille
sekä generoi näkymät. Mahdollistaa suorituksen poiston, suorituksen tallentamisen, suorituksen pisteiden muuttamisen sekä tulostaa suorituksen muokkaussivun.

AuthenticationController.java:
Toteuttaa AuthenticationControllerInterfacen määrittelemän rajapinnan. Kontrolleri kuuntelee

autentikaatioon liittyviä pyyntöjä. Mahdollistaa rekisteröitymisen, sisäänkirjautumisen, uloskirjautumisen. Lisäksi tulostaa pääsivun(mainpage.jsp), rekisteröitymissivun ja sisäänkirjautumissivun.

LeaderboardController.java:
Toteuttaa LeaderboardControllerInterfacen määrittelemän rajapinnan. Kontrolleri kuuntelee leaderboardiin liittyviä pyyntöjä ja ohjaa ne eteenpäin serviceille sekä generöi näkymiä. Mahdollistaa leaderboardin luonnin, leaderboardin poiston, käyttäjän lisäämisen leaderboardiin. Generoi leaderboardnäkymän ja leaderboardin editointinäkymän.

UserController.java:
Toteuttaa UserControllerInterfacen määrittelemän rajapinnan. Kontrolleri generoi käyttäjäsivun sekä käyttäjän kontrollointi ja infosivun.

Helpers:
ConnectionHelper.java:
Apuluokka, joka luo yhteyden tietokantaan.

Repository:
AccomplishmentSQLRepo.java:
Luo Accomplishment-taulun kantaan kun ohjelma käynnistetään, sisältää lisäksi SQL:t suorituksen luomiseen, pisteiden tallentamiseen, suorituksen hakemiseen, kaikkien suoritusten hakemiseen, suorituksen poistamiseen ja käyttäjän kaikkien suoritusten hakemiseen

LeaderboardSQLRepo.java:
Luo Leaderboard-taulun ja Leaderboard_Users-taulun kun ohjelma käynnistetään. Sisältää lisäksi SQL:t käyttäjän lisäämiseen leaderboard_users tauluun, kaikkien leaderboardin userien hakemiseen, kaikkien käyttäjän leaderboardien hakemiseen, leaderboardin luomiseen, leaderboardin hakemiseen, kaikkien leaderboardin hakemiseen ja leaderboardin poistamiseen

UserSQLRepo.java
Luo User-taulun kun ohjelma käynnistetään. Sisältää SQL:t userin luonnille, sisäänkirjautumisen tarkistukselle, kaikkien userien haulle, yhden userin haulle ja userin pisteiden päivitykselle.

Service:
AccomplishmentService.java:
Toteuttaa AccomplishmentServiceInterface rajapinnan, vastaanottaa Controllerin pyyntöjä, muodostaa niistä kantaan soveltuvia osia ja antaa nämä osat sitten repoille. Mahdollistaa suorituksen luonnin, pisteiden tallennuksen, suorituksen hakemisen, kaikkien suoritusten hakemisen, suorituksen poistamisen, käyttäjän suoritusten hakemisen ja Accomplishment-taulun luonnin.

LeaderboardService.java:
Toteuttaa LeaderboardServiceInterface rajapinnan, vastaanottaa Controllerin pyyntöjä, muodostaa niistä kantaan soveltuvia osia ja antaa nämä osat sitten repoille. Mahdollistaa Leaderboard-taulun luonnin, Leaderboard_Users-taulun luonnin, leaderboardien luomisen, leaderboardin haun, kaikkien leaderboardien haun, leaderboardin poiston, käyttäjien lisäämisen leaderboardiin, leaderboardin käyttäjien hakemisen ja

käyttäjien leaderboardien hakemisen.

UserService.java:
Toteuttaa UserServiceInterface rajapinnan, vastaanottaa Controllerin pyyntöjä, muodostaa niistä kantaan soveltuvia osia ja antaa nämä osat sitten
repoille. Mahdollistaa rekisteröinnin, roolin vaihtamisen, sisäänkirjautumisen tarkistamisen, User-taulun luonnin, adminUserin luonnin, tavallisen userin
luonnin, userin haun, kaikkien usereiden hakemisen ja userin pisteiden päivittämisen.

9. Asennustiedot

Sovellus on nyt asennettu Herokulle osoitteeseen: blooming-citadel-2401.herokuapp.com/app/mainpage
Sovelluksen saa asennettua helposti uuteen ympäristöön ottamalla githubista clonen ja buildaamalla mavenilla projektin. Tämän jälkeen projektin voi ajaa
esimerkiksi jettyllä suoraan komennolla: java -jar target\dependency\jetty-runner.jar target*.war (windows ympäristössä, linuxissa kenoviivat toistepäi)
Tietokannan asetukset määritellään src/main/webapp/WEB-INF kansiosta löytyvään database.xml:n

10. Käynnistys-/Käyttöohje

Järjestelmä löytyy osoitteesta: blooming-citadel-2401.herokuapp.com/app/mainpage
Admintunnukset ovat: admin adminPwd
Ohjelman käytön pitäisi olla selkeää, etusivulla on kaikki leaderboardit, nimeä painamalla pääsee tarkastelemaan leaderboardia, navigaatiopalkissa on linkit
käyttäjänhallintaan, suoritustenhallintaan ja leaderboardienhallintaan.

11. Testaus, tunnetut bugit ja puutteet & jatkokehitysideat

Ohjelmassa ei ole yhtään automaattista testiä, joka on sen selvä heikkous. Jos ohjelmaa lähdettäisiin jatkokehittämään kannattaisi aloittaa automaattisesta
testauksesta. Olen testannut ohjelmaa antamalla lomakkeille erilaisia syötteitä ja katsonut meneekö ohjelma rikki. Minkään syötteen ei pitäisi rikkoa ohjelmaa.
Lisäksi olen testannut ohjelmaa xss-scriptejä vastaan.
Ohjelmaa voisi kehittää eteenpäin lisäämällä erilaisia joukkueita mihin käyttäjät voisivat kuulua ja sitten lisätä näille joukkueille omia leaderboardeja,
joissa joukkueet kilpailisivat vastakkain.
Ainut bugi minkä tiedän on se, että on mahdollista lisätä kaksi käyttäjää samalla käyttäjänimellä.

12. Omat kokemukset

Tietokantasovelluksessa helppoa oli perus mvc-toiminnallisuuden rakentaminen. Olen ennenkin tehnyt vastaavanlaisia ohjelmia Springillä jolloin Controllerien,
Serviceiden ja mallien tekeminen oli helppoa. Suurimmat vaikeudet olivat Springin konfiguroinnissa, kantayhteyksien konfiguroinnissa ja tietokannan
miettimisessä erityisesti se miten linkkaan käyttäjät ja leaderboardit yhteen, koska aikaisempaa

tietokantojen suunnittelukokemusta ei paljoa löytynyt.