Joona Nuorala
Kalle Vesanterä

FRISBEEGOLF-TULOSALUSTAN OHJELMISTOTUOTANTOSUUNNITELMA

Ohjelmistotuotanto
CENTRIA-AMMATTIKORKEAKOULU
Tieto- ja viestintätekniikan insinööri
Joulukuu 2024

SISÄLLYS

1 PROJEKTISUUNNITELMA JA ROOLITUS

1.1 Tavoitteet

Projektin tavoitteena on luoda uudenlainen alusta, jota käytetään frisbeegolf-turnauksissa ja peleissä. Vastaavanlaisia palveluita on jo olemassa, mutta suunnittelemaamme täysimittaista alustaa, joka toimii sekä täysimittaisena web-sovelluksena että natiivina mobiilisovelluksena ei ole olemassa.

Suurimpana tavoitteena onkin tavoittaa entistä suurempi yleisö, kuten pelaajat, kisajärjestäjät ja ratojen ja kilpailujen suunnittelijat ja järjestäjät.

Alustan on tarkoitus mahdollistaa sekä suurempien turnausten järjestäminen, että pienempien kaveriporukoiden väliset hupipelit. Mobiilisovellusta käyttäessä pelaaja voi helposti syöttää palveluun pelinaikaista dataa, kuten heitossa käytetyn kiekon ja kiekon sijainnin. Muut pelaajat voivat samalla seurata muiden pelaajien edistymistä esimerkiksi sovelluksen sisäisen karttapalvelun avulla.

Pelin aikainen data on saatavilla reaaliaikaisesti myös täysimittaisessa web-sovelluksessa, josta esimerkiksi kisajärjestäjät voivat seurata turnauksen edistymistä.

Kaikki pelien aikainen data myös tallennetaan, jotta esimerkiksi pelaajat voivat käydä tarkastelemassa heidän pelattuja turnauksiaan.

1.2 Roolitus ja vastuut

Roolitus on suunnitteluvaiheessa jaettu seuraavan kaltaiseksi. Ohjelmistotuotannon alkaessa tulee ohjelmistotuotantosuunnitelmaan päivittää mahdolliset muutokset.

Kalle Vesanterän vastuulla on olla teknisen puolen ”asiantuntija” joka selvittää mitä teknisiä vaatimuksia ohjelmiston eri osilla tulee olemaan. Käytännössä tämä myös tarkoittaa sitä, että Kallen vastuulla on ohjelmiston backend ja sen suunnittelu.

Joona Nuoralan vastuulla on toimia eräänlaisen ”loppukäyttäjä-asiantuntijana”, eli tarkoitus on ideoida ominaisuuksia sekä sitä miten UI/UX puoli tulee toimimaan sovelluksessa. Tämä siis vastaa ohjelmiston frontend-puolta ja suunnittelua.

Molemmat tekevät oman osa-alueensa alustavaa dokumentaatiota. Yhteisissä tapaamisissa myös käydään tarkemmin läpi mitä suunnitellut ohjelmiston palat sisältävät, samalla myös sovittaen dokumentaatiota yhteen.

Koska kysymys on pienestä tiimistä, tullaan työtä tekemään myös paljon yhdessä. Asioita sekä frontend- sekä backend-asioissa voidaan päättää yhdessä, mutta epäselvyyksien välttämiseksi pääasiallinen vastuu on näiden osa-alueiden omistajilla.

1.3 Yhteydenpito

Projektia tehdään pienessä ryhmässä ja työ tulee olemaan pääasiassa etätyöskentelyä. Yhteydenpito on suunniteltu tapahtuvan niin, että yhteisiä tapaamisia on tiiviisti ja näiden aikana katsotaan tarkasti mitä projektissa on tapahtunut, samalla sopien tulevasta.

Jatkuvaa yhteydenpitoa varten käytämme parhaaksi katsomaamme sähköistä viestintää, joka mahdollistaa niin teksti-, kuin ääni- ja videomuotoisen yhteydenpidon. Lisäksi yhteydenpitoon liittyy oleellisesti työkalut kuten Trello, Github ja muut työkalut joita projektin aikana käytetään.

1.4 Prosessimallin valinta

Prosessimalliksi valittiin inkrementaalisen- ja prototyyppimallin hybridi. Tämä päätös tehtiin sen takia, että pienellä tiimillä toimiessa, sekä täysin uudenlaisen projektin kanssa työskennellessä, on tämän kaltainen malli helpompi hallita. Lisäksi tämän mallin myötä ei kulu aikaa turhaan suunnitteluun ja erilaisiin kokouksiin, vaan työn pääpainona voidaan pitää itse ohjelmiston parissa työskentely. Koska kyseessä on myös tiimin ensimmäinen projekti, prototyyppimainen komponenttien kehittely ja kokeilu on huomattavasti hedelmällisempää. Täten saamme itse kehitettyä projektin sellaiseksi kuin haluamme, varaten kuitenkin mahdollisuuden muuttaa asioita ikään kuin lennosta.

Olemme myös itse projektimme täysimittaiset omistajat, joten myös sen takia tämä malli sopii projektiin parhaiten.

1.5 Tehtävät ja aikataulu

Alustavasti tehtävät on määritelty seuraavalla tavalla:

- Projektin ideointi, alustavan projektisuunnitelman tekeminen ja roolien valinta
- Ohjelmiston vaatimusmäärittely, joka sisältää mm. toiminnalliset ja ei-toiminnalliset vaatimukset, ominaisuuksien priorisoinnin ja käyttäjiin ja heidän tarpeisiinsa liittyvät vaatimukset
- Ohjelmiston arkkitehtuurien ja moduulien suunnittelu, sisältäen frontend- ja backend-toiminnallisuuden, rajapinnat, riippuvuudet ja muut tekniset yksityiskohdat. Näistä koostuu myös koko järjestelmän kuvaus.
- CI/CD-suunnittelu, joka sisältää tuotteen- ja versionhallinnan sekä testauksen ja laadunvarmistuksen suunnitelmat

Aikataulu:
Projektin suunnitelluksi aikajanaksi on arvioitu seuraavaa:

- Projektin suunnittelu on parhaillaan käynnissä ja suunnitelman esityskelpoiseksi valmistumisen määräaika on 16.12.2024
- Ohjelmistokokonaisuuden rakentaminen alkaa 6.1.2025
- Ohjelmistokokonaisuuden odotetaan olevan valmis ensimmäiseen tuotantoversioon (alpha) 22.12.2025
- Lisäksi projektisuunnitelman kehittyessä, tulemme arvioimaan tarkempia aikajanoja ohjelmiston eri osille ja niiden määräajoille. Nämä tullaan päivittämään suunnitelmaraporttiin niiden selkeytyessä.

1.6 Riskianalyysi

Projektille on kirjoitushetkellä tehty alustavaa riskiarviota. Tämä on aihe joka tulee myös elämään projektin kasvaessa ja sitä päivitetään tapauskohtaisesti reaaliajassa.

Tällä hetkellä riskiarvio sisältää seuraavat kohdat ja niiden todennäköisyydet ja vaikutukset:

• Taloudellinen riski (kustannuksia ei saada katettua) – Riski on pieni, sillä projektiin ei nykytiedon mukaan tulla käyttämään suuria taloudellisia panostuksia. Vaikutus on myös minimoitu, sillä se ulottuu ainoastaan projektin sisäisiin omistajiin. Vaikutuksia saadaan myös hallittua tekemällä harkittuja päätöksiä hankinnoissa ja pitämällä kustannukset mahdollisimman pieninä.

• Käyttäjäkunta ei koskaan kasva halutuksi – Riski on kohtalainen. Tähän vaikuttaa suhteellisen pieni kohderyhmä ja se että olemassa olevia palveluita on markkinoilla. Toisaalta alustan onnistuessa halutusti, voi tulos ottaa tuulta alleen suhteellisen nopeasti. Vaikutus on vakava, sillä käytännössä tässä kohtaa projekti on epäonnistunut. Riskin toteutuessa on mahdollista katsoa, voiko mahdollisesti epäonnistuneen projektin osia käyttää jossakin tulevassa projektissa, jotta työ ei olisi ollut täysin turhaa. Lisäksi tämän kaltainen projektin kaatuminen ei välttämättä ole epäonnistuminen, sillä kyseessä on kuitenkin arvokasta tietoa jota voidaan hyödyntää myöhemmin uralla, varsinkin kun suurta taloudellista panostusta ei ole tehty.

• Ryhmän jäsenen sairastuminen tai äkillinen poisjäänti – Riski on pieni, mutta tämän vaikutukset voivat olla katastrofaalisia ja projekti voi viivästyä tai jopa kokonaan keskeytyä tämän johdosta. Tämä on myös asia, johon ei käytännössä voi aina vaikuttaa. Riskin toteutuessa vaikutuksia voidaan minimoida sillä, että kaikki tiimin jäsenet ovat perillä projektin eri osa-alueista ja toimintatavoista. Tällä voidaan mahdollistaa se, että projekti ei täysin pysähdy, vaikkakin aikataulutus saattaa muuttua radikaalisti. Lisäksi tehokas viestintä on suuri osa tämän kaltaisen riskin minimointia.

• Kehityslaitteiston tai infra-laitteiston hajoaminen – Riski on pieni ja vaikutukset ovat pienet, jos ne on huomioitu projektin kulussa. Tässä tulee kuitenkin huomioida mahdollinen taloudellinen riski. Riskin vaikutuksia voidaan hallita esimerkiksi pitämällä tarpeelliset varmuuskopiot kaikesta projektiin liittyvästä. Lisäksi päätös ostaa laitteistopuolen palvelut ulkoiselta toimijalta takaa pitkälti sen, että laitteiston pettäminen ei vaikuta projektin kulkuun.

• Aikataulutuksen pettäminen – Riski on kohtalainen. Kyseessä on tiimin ensimmäinen vastaavanlainen projekti, jolloin riski virheelliseen aikataulutukseen kasvaa. Lisäksi yllättäviä muutoksia saattaa tapahtua, joka vaikuttaa projektin kulkuun suunnitellussa aikataulussa. Riskiin voi vaikuttaa valmistautumalla etukäteen mahdollisiin tilanteisiin, jotka tähän vaikuttaa, sekä myös

laskemalla aikataulun eri osiin tarpeeksi ylimääräistä aikaa. Lisäksi projektin aikainen työajanseuranta helpottaa aikataulutuksen arviointia projektin edetessä, jolloin riski tämän toteutumiseen pienenee.

• Jos projekti ei ole päätoimista toimintaa, miten ajanhallinta saadaan toimimaan – Tähän sisältyy monia riskejä ja niiden vaikutukset voivat olla erilaisia. Parhaimmassa tapauksessa vaikutus on minimaalinen mutta pahimmillaan projekti voidaan joutua keskeyttämään.

• Kyberturvariskit – Kyberturva on osa-alue, joka tulee ottaa tosissaan, sillä riskit voivat olla hyvinkin suuret. Esimerkiksi ransomware-hyökkäyksen kohteeksi joutuminen voi olla kohtalokasta projektille. Tähän voidaan kuitenkin varautua noudattamalla hyviä toimintatapoja projektin datan hallinnassa, laitteiston ja ohjelmistojen käsittelyssä sekä yleisessä toiminnassa, jota projekti sisältää.

2 VAATIMUSMÄÄRITTELY

2.1 Toiminnalliset ja ei-toiminnalliset vaatimukset, priorisointi

Toiminnalliset ja ei-toiminnalliset vaatimukset määriteltiin pääosin projektin omistajien, eli kehittäjien, toimesta. Vaatimuksia käytiin läpi suunnittelupalavereissa. Niissä hyödynnettiin omia ideoita ja visiota alustaa kohtaan, alustavia käyttäjäkyselyjä sekä vastaavanlaisten alustojen käyttäjäkokemuksesta syntyneitä ideoita ja tarpeita.

Lisäksi vaatimusten priorisointia mietittiin vaatimuksia suunnitellessa. Priorisoinnissa pidettiin tärkeimpänä niitä ominaisuuksia, jotka koemme välttämättömäksi alustan suunnitellun toiminnallisuuden kannalta. Vaatimuksissa on myös ominaisuuksia, pienemmällä prioriteetilla, joiden koemme tuovan alustalle lisää syvyyttä. Nämä eivät kuitenkaan ole kriittisen tärkeitä alustalle kokonaisuus huomioiden.

Kokonaisuudessaan luettelo vaatimuksista ja ei-toiminnallisista vaatimuksista priorisointeineen löytyy taulukkona suunnitelman liitteistä. (LIITE 1)

2.2 Vaatimusten validointi

Ohjelmiston suunnittelun visiointivaiheessa on tehty kattava lista toiminnallisista ja ei-toiminnallisista vaatimuksista, jotka kuvaavat sitä millaiseksi ohjelmisto tulee rakentaa. Nämä vaatimukset validoidaan ja verifioidaan suunnitteluvaiheessa, sekä uudelleen juoksevasti kun ohjelmistoa rakennetaan. Lisäksi ennen ohjelmiston tuotantoon saattamista, validoidaan ja verifioidaan rakennetun ohjelmiston sisältö. Tällä tavoin varmistutaan siitä, että rakenteilla oleva ohjelmisto on alkuperäisen vision ja tarkoituksen mukainen.

Vaatimusten validointia tehdään yhdessä kaikkien sidosryhmien kanssa. Koska ohjelmiston omistajat ovat ohjelmiston tuottajia, on heillä kuitenkin suurin ja viimeinen päätäntävalta ohjelmiston sisältöön ja ominaisuuksiin.

Vaatimuksiin tehdyt muutokset:

- Suunnitteluvaiheen vaatimusten validointi 13.11.2024
  - SP-5/001 – Päivitetty tarkempaan kuvaukseen
  - SP-9/003 – Päivitetty korkeampaan prioriteettiluokkaan

2.3 Webropol-kysely

Vaatimusmäärittelyä varten teimme alan harrastajille suunnatun ytimekkään kyselyn, jossa kartoitimme tärkeimpiä ominaisuuksia, joita ohjelmiston tulisi sisältää. Kirjoitushetkellä kysely oli avattuna auki n. 4 päivää, yhden viikonlopun yli. Vastauksia kyselyyn oli tullut 16. Kyselyn alkuperäinen Webropol-raportti tämän suunnitelman liitteenä. (LIITE 2)

Kyselyssä kartoitettiin seuraavia asioita:

- Mitä frisbeegolfiin liittyviä palveluja olet käyttänyt?

  100% vastaajista on käyttänyt Disc Golf Metrix -palvelua. Lisäksi PDGA Live sekä UDisc ovat olleet suosittuja alan harrastajien keskuudessa. Näihin ohjelmistoihin kannattaa siis kiinnittää erityisesti huomiota uutta suunnitellessa.

- Mistä seuraavista ominaisuuksista olet eniten kiinnostunut frisbeegolf-tulosalustoissa?

  Tässä kysymyksessä pyydettiin valitsemaan 1-4 ominaisuutta, jotka ovat tärkeimmät alustalle. Yli 50% kannatukselle ylsi 3 vaihtoehtoa:

  1. Mahdollisuus selata omaa peli- ja turnaushistoriaa
  2. Mahdollisuus luoda omia kilpailuja, esimerkiksi muutaman hengen kaveriporukalle
  3. Kilpailun tulosten reaaliaikainen seuranta

  Lisäksi noin 40% kannatuksesta oli seuraaville ominaisuuksille:

  4. Karttanäkymä lähialueen radoista
  5. Erilaisia valmiita pelimuotoja (kuten paripeli ja skins)

  Yllättäen, karttaominaisuus, jossa voi seurata kilpailun etenemistä, ei tässä osassa kyselyä saavuttanut suurta suosiota. Kyse on kuitenkin yhdestä ydinominaisuudesta, jota alustalle on suunniteltu, joten sille saattaa olla kannattavaa antaa joka tapauksessa suurempi prioriteetti.

- Millaisista lisäominaisuuksista olet kiinnostunut?

  Lisäominaisuuksista kyseltäessä annettiin vastaajien valita vapaasti niin monta vaihtoehtoa kuin he halusivat. 3 vastausta ylsi 50% kannatusrajaan:

  1. Heittopituuden mittaaminen
  2. Ratakohtainen Highscore-taulukko
  3. Tuloskortin jakaminen esim. sosiaalisessa mediassa

  Nämä ovat ominaisuuksia, jotka kannattaa todennäköisesti nostaa ohjelmiston kokonaisuus huomioon ottaen melko korkealle prioriteeteissa.

  Noin 40% kannatuksen keräsivät myös 2 lisäominaisuutta:

  4. Puttiharjoitus
  5. Radalla käytetyn reitin tallentaminen karttapalveluun

  Nämä ominaisuudet yhdistettynä alustaan suunniteltuun karttapalveluun toimivat todennäköisesti melko hyvin yhteen. Tämän takia nämä kannattaa myös huomioida ohjelmiston kokonaisuutta toteutettaessa, vaikkakaan niiden prioriteettia ei tarvitse alkuun pitää kovin korkealla.

- Millä seuraavista alustoista olisit kiinnostunut käyttämään sovellusta?

  100% vastaajista on halukkaita käyttämään palvelua mobiililaitteella, kuten älypuhelimella. Lisäksi 38% kertoi olevansa kiinnostuneita käyttämään palvelua tietokoneella. Tämä viestii siitä, että alustan mobiiliversion kehittäminen kannattaa nostaa prioriteettijärjestyksessä mahdollisimman korkealle.

- Mitä ominaisuuksia olet jäänyt kaipaamaan olemassa olevissa palveluissa?

  Tähän kysymykseen saimme seuraavat vastaukset:

  1. *”Esim metrixissä voisi olla tulostenkirjausnäkymässä sijoitusseuranta, esim. niin, että näkyy omaa sijaa 4 lähintä pelaajaa ja heidän kokonaistulos.”*
  2. *”Selkeä ja yksinkertaistettu kisaluettelo jotka saa järjestettyä aika järjestykseen milloin ilmoittautuminen aukeaa. Tai karttapalvelu jossa näät kisat jotka on avoinna ilmottautumiselle.”*
  3. *”Karttoja”*

  Alustavasti suunniteltu karttapalvelu vaikuttaa siis olevan ominaisuus, jolle olisi kysyntää, muodossa tai toisessa.

- Vapaa ideoiden antaminen.

  Tähän kysymykseen saimme seuraavat vastaukset:

  1. *” Selkeä ja yksinkertaistettu kisaluettelo jotka saa järjestettyä aika järjestykseen milloin ilmoittautuminen aukeaa. Tai karttapalvelu jossa näät kisat jotka on avoinna ilmottautumiselle”*
  2. *”Tarkempaa dataa radoista.”*

2.4 User story -esimerkit

Yksi osa User story -esimerkkien keräämisessä on tekemämme Webropol-kysely ohjelmiston kohderyhmän käyttäjille. Lisäksi User storyjä kehitettiin itsenäisesti ohjelmistosuunnittelun aikana, jotta saatiin käsitys siitä, miten alustan suunnitellut ominaisuudet tukevat käyttäjäkokemusta.

Alla on listattuna User storyjä, liittyen ohjelmiston tiettyihin ominaisuuksiin:

1. Rekisteröityminen - Käyttäjänä haluan mahdollisuuden rekisteröityä sivustolle, jotta voin tallentaa tietojani palveluun ja katsella niitä jälkikäteen.

2. Salasanan nollaus - Käyttäjänä haluan mahdollisuuden nollata käyttäjätunnuksen salasanan, jos vaikkapa unohdan salasanani.

3. PDGA-numero käyttäjätilissä - Käyttäjänä haluan mahdollisuuden tallentaa PDGA-numeron käyttäjälleni, jotta PDGA-rating saadaan näkyviin palvelussa.

4. Tilin poistaminen - Käyttäjänä haluan mahdollisuuden poistaa käyttäjätili koska tahansa palvelusta, ilman mitään syytä, sillä haluan säilyttää päätösvaltani omien tietojeni käytöstä.

5. Profiilin tietojen julkisuus - Käyttäjänä haluan päättää itse mitkä tiedot profiilissani ovat julkisia, jotta voin päättää itse mahdollisesti yksityisyyteen vaikuttavien tietojen julkisuudesta.

6. Karttanäkymä lähialueen radoista - Käyttäjänä haluan nähdä itseäni lähellä olevat radat helpossa karttanäkymässä, jotta löydän reitin radoille jopa vieraalla paikkakunnalla ollessani.

7. Kilpailuihin liittyminen - Käyttäjänä haluan nähdä helposti listattuna kilpailut, joiden rekisteröityminen on auki.

8. Kilpailuihin liittyminen - Käyttäjänä haluan nähdä kartalla tulevat kilpailut, jotta voin helposti osallistua minua lähellä järjestettäviin kilpailuihin.

9. GPS-sijainnin käyttö - Käyttäjänä haluan mahdollisuuden käyttää puhelimen sijaintitietoja, jotta voin tallentaa esimerkiksi heittojen pituuksia ja radalla käytetyn reitin.

10. Tulosten kirjaaminen - Käyttäjänä haluan, että tuloksien kirjaaminen on mahdollisimman helppoa ja nopeaa, jolloin voin keskittyä paremmin pelin muihin osa-alueisiin.

11. Pelihistoria - Käyttäjänä haluan nähdä oman pelihistoriani, jotta voin seurata kehittymistäni ajan mittaan.

12. Heittotekniikan harjoittelu - Käyttäjänä haluan mahdollisuuden mitata heittojeni pituutta, tallentaen sen historiaan, jotta voin seurata kehittymistäni.

13. Sosiaaliseen mediaan jakaminen - Käyttäjänä haluan jakaa tuloskortin sosiaaliseen mediaan, esimerkiksi hyvän kierroksen pelattuani, jotta ystäväni pääsevät näkemään hyvän tuloksen.

14. Turnausten hallinta - Turnausjärjestäjänä haluan mahdollisuuden lisätä ja poistaa pelaajia turnauksesta, jotta hallinta olisi mahdollisimman helppoa.

15. Turnausten hallinta - Turnausjärjestäjänä haluan mahdollisuuden lisätä käyttäjiä turnauksen hallintaryhmään, jotta vaikkapa tuomarit saadaan osaksi turnauksenhallintaa.

16. Turnausten hallinta – Turnauksen järjestäjänä haluan mahdollisuuden lähettää osallistuville pelaajille viestejä, jotta kommunikaatio pelaajien kanssa olisi mahdollisimman sujuvaa.

17. Turnausten hallinta – Turnauksen järjestäjänä haluan mahdollisuuden lisätä turnaukseen eri luokkia, jotta esimerkiksi eri ikäluokat voidaan jakaa keskenään omiin peleihinsä.

18. Turnausten hallinta – Turnauksen järjestäjänä haluan mahdollisuuden jakaa pelaajat eri ryhmiin, jotta esimerkiksi mahdollisen sponsorin vaatimukset saadaan täytettyä.

19. Alustan käyttö - Käyttäjänä haluan mahdollisuuden selata alustan ohje-sivua, tai vastaavaa, jotta löydän tietoa ja apua siitä miten ominaisuuksia tulee käyttää.

20. Säännöt - Käyttäjänä haluan mahdollisuuden nähdä frisbeegolfin säännöt ja turnauksen sisäiset säännöt, jotta mahdollisilta väärinkäsityksiltä vältyttäisiin.

2.5 Projektin sidosryhmät

Ohjelmistokokonaisuuden sidosryhmiksi on määritelty seuraavat:

1. Ohjelmiston omistaja

   Ohjelmiston omistajaksi on määritelty ohjelmiston tuottava ryhmä. Tämä perustuu siihen, että tarkoituksena on tuoda markkinoille uusi ohjelmisto, joka on täysin itse kehitetty ja tullaan tuomaan omasta toimesta markkinoille.

1. Toimittaja

   Ohjelmiston toimittajaksi on määritelty ohjelmiston tuottava ryhmä. Tuomme itse ohjelmiston markkinoille ja vastaamme sen julkisaattamisesta.

2. Tukiryhmä

   Tämä sidosryhmä kattaa kaikki ohjelmiston kehitykseen mahdollisesti osallistuvat ulkopuoliset asiantuntijat. Esimerkiksi ulkopuolelta palkattu konsultti tai tietoturva-asiantuntija kuuluu tukiryhmään.

3. Ohjelmistotestaajat

   Esimerkiksi ohjelmiston tuotannon alpha- ja betavaiheessa voidaan ohjelmistoa testata kutsumalla rajattu määrä käyttäjiä käyttämään ohjelmistoa. Heitä käsitellään omana sidosryhmänään, joka tuottaa palautetta ohjelmiston toiminnasta jatkokehittämistä varten.

4. Käyttäjät

   Ohjelmiston loppukäyttäjät ovat todennäköisesti kaikista laajin sidosryhmä. Tämä kategoria kattaa kaikki lopullisen ohjelmiston käyttäjät. Siihen sisältyy niin yksittäiset pelaajat, kuin myös mahdollisten kilpailuiden järjestäjät.

3 ARKKITEHTUURI- JA MODUULISUUNNITTELU, SAAVUTETTAVUUS

3.1 Koko järjestelmän kuvaus

Ohjelmistokokonaisuus on jaettu kahteen pääosaan, frontendiin sekä backendiin. Nämä molemmat koostuvat useammista pienistä komponenteista, jotka voivat oman pääosansa sisällä toimia yhteistyössä keskenään. Lisäksi frontend ja backend voivat keskustella toisiensa kanssa ohjelmistorajapintojen kautta.

Frontend-kokonaisuus koostuu kahdesta isosta osasta; web-pohjaisesta ja mobiililaitteille tarkoitetusta frontendistä. Nämä tulevat pitämään sisällään pitkälti saman toiminnallisuuden, mutta ne tulee kuitenkin rakentaa erikseen. Tämän työn helpottamiseksi ja nopeuttamiseksi, on näiden toteutusteknologioiksi valittu React ja React Native, joista ensimmäinen on tarkoitettu web-pohjaiseen frontendiin ja jälkimmäinen mobiililaitteiden frontendiin. Frontend koostuu monesta eri pääkomponenteista, jotka lähestulkoon kaikki voivat olla yhteydessä toisiinsa. Erilaisia komponentteja (tai näkymiä) yhdistellessä saadaan aikaan kokonaisuus, joka toimii yhteen saumattomasti ja yhteneväisesti.

Backend käsittää kaiken sen, mitä sovellus tekee taustatyönä. Käyttäjien-, turnausten- ja karttojenhallinta frontendissä saa tarvitsemansa tiedot backendin tietokannoista, rajapintojen kautta. Nämä tiedot ovat tallennettuna omiin tietokantoihinsa. Lisäksi tietoja muokattaessa frontendin puolella, välitetään data backendin rajapintojen kautta tietokantoihin. Lisäksi ulkoisista palveluista haetaan data backendissä. Esimerkiksi pelaajan tietoihin haettava tieto PDGA:n rajapinnoista, käsitellään backendissä, jonka jälkeen tämä tallennetaan vastaavaan tietokantaan. Myös erilaisten notifikaatioiden, kuten sähköpostien ja push-notifikaatioiden, lähettäminen tapahtuu backendin kautta.

Ohjelmisto on kokonaisuutena suunniteltu toimimaan pilvipohjaisessa Platform-as-a-Service (PaaS) -palvelussa. Tällä tavalla kehittäjien ei tarvitse itse huolehtia esimerkiksi laitteistopuolen hankinnoista ja ylläpidosta. Lisäksi pilvipohjaisissa järjestelmissä on pitkälti sisäänrakennettuna tietoturvapuolen ominaisuuksia, kuten myös nopeaa skaalautumista tukevia ratkaisuja. Lisäksi ohjelmistovaatimus ohjelmiston saatavuudesta saadaan varmemmin täytettyä pilvipohjaisella ratkaisulla. Web-pohjainen frontend ja backend tullaan toteuttamaan PaaS-palveluun omina instansseinaan, jotka toimivat yhteistyössä keskenään.

Liitteistä löytyvät kaaviot kuvaavat järjestelmän toimintaa ja sitä, miten kokonaisuuden eri komponentit viestivät toistensa kanssa. (LIITE 3)

3.2 Ositus

Järjestelmän pääosat, backend ja frontend, on jaettu useampaan pienempään komponenttiin. Näiden komponenttien osittaminen, eli yksittäisen kokonaisuuden muodostavat komponentit, löytyvät tästä osasta kirjallisessa muodossa. Nopean pikakatsauksen komponenteista saa liitteissä olevista kaavioista. (LIITE 3)

1. Backend
    - Käyttäjänhallinta
        - Rajapinta frontendin ja backendin välillä
        - Backend-keskuskomponentti
        - Rajapinta keskuskomponentilta tietokantaan
        - Käyttäjienhallinnan tietokanta

    - Tulospalvelun datankäsittely
        - Rajapinta frontendin ja backendin välillä
        - Backend-keskuskomponentti
        - Rajapinta keskuskomponentilta tietokantaan
        - Tietokanta tulospalvelun datalle

    - Karttapalvelun datankäsittely
        - Rajapinta frontendin ja backendin välillä
        - Backend-keskuskomponentti
        - Rajapinta keskuskomponentilta tietokantaan
        - Tietokanta karttapalvelun datalle

- Notifikaatioiden lähettäminen
  - Sähköposti
    - Backend-keskuskomponentti
    - Moduuli sähköpostien lähettämistä varten
    - Rajapinta moduulin hallintaan
  - Push-notifikaatiot
    - Backend-keskuskomponentti
    - Moduuli push-notifikaatioiden lähettämistä varten
    - Rajapinta moduulin hallintaan

- Ulkoisten palveluiden kanssa viestiminen (PDGA:n rajapinta)
  - Backend-keskuskomponentti
  - Moduuli PDGA:n rajapinnan kanssa viestimiseen
  - Rajapinta käyttäjienhallinnan tietokantaan

2. Frontend
   - Navigaatio frontendissä
     - Käyttönäkymään elementti, jossa skaalautumisen mukaan:
       - Suurella skaalauksella: Elementti jossa vierekkäin listattuna painikkeet, joista päästään navigoimaan sisäisesti sovelluksessa
         tai
       - Pienellä skaalauksella: Elementti jossa painike, jota painamalla avautuu lista painikkeista, joista päästään navigoimaan sisäisesti sovelluksessa
         tai
       - Mobiililaitteilla: Elementti, jossa pyyhkäisemällä voidaan karusellimaisesti selata painikkeita, joista päästään navigoimaan sisäisesti sovelluksessa

- Kirjautumis- ja rekisteröitymisnäkymä
  - Näkymä, jossa painikkeet:
    - Kirjautuminen
      - Siirrytään näkymään jossa lomake:
        - Käyttäjänimi/sähköpostiosoite
        - Login-nappula
        - Unohtuneen salasanan palautusmahdollisuus
    - Rekisteröityminen
      - Siirrytään näkymään jossa lomake:
        - Käyttäjänimi
        - Sähköpostiosoite
        - Salasana
        - Salasanan varmistus
        - Rekisteröidy-nappula
        - Palaa takaisin edelliselle sivulle -nappula
    - Frontendin koodissa rajapintayhteys backendiin

- Profiilinäkymä
  - Näkymä, jossa seuraavanlaiset elementit:
    - Käyttäjänimi
    - Oikea nimi (jos käyttäjä sallii)
    - Profiilikuva
    - Kotimaan lippu ja lyhenne (jos käyttäjä sallii)
    - PDGA-numero (valinnainen)
    - Pelaajan peleissä käyttämät kiekot, in-the-bag
    - Statistiikkaelementti, jossa koostettuna pelaajan pelihistorian tietoja
    - Painike käyttäjälle, josta oman profiilin tietoja voi muokata
  - Frontendin koodissa rajapintayhteys backendiin

- Ratojen listaus
  - Näkymä, jossa seuraavanlaiset elementit:
    - Listaus radoista, jotka ovat järjestetty etäisyyden mukaan
      - Tarkka sijainti vaatii yhteyden käyttäjän geo-lokaation
    - Ratakohtainen alasvetovalikko, jossa tarkempia tietoja ko. Radasta
      - Painike, josta avautuu ko. radan kaikki tiedot
  - Frontendin koodissa rajapintayhteys backendiin

- Tulosnäkymä
  - Näkymä, jossa seuraavanlaiset elementit ja ominaisuudet:
    - Listaus kilpailun osallistujista
      - Top 3 kilpailijat korostettuna
      - Jos käyttäjä on itse osallistuja kilpailussa, korostetaan käyttäjän sijainti listauksessa muihin nähden
    - Kilpailija valittuna, näytetään lisää tietoa ko. kilpailijasta
      - Väyläkohtaiset tulokset
      - Kierroskohtaista statistiikkaa
  - Frontendin koodissa rajapintayhteys backendiin

- Kilpailunhallinta
  - Näkymä, jossa seuraavanlaiset elementit ja ominaisuudet:
    - Lista jo olemassa olevista kilpailuista
      - Mahdollisuus liittyä johonkin näistä kilpailuista
      - Jokaisella kilpailulla oma infonäkymä
      - Jos käyttäjä on kilpailun admin, mahdollisuus siirtyä ko. kilpailun hallintanäkymään
    - Mahdollisuus luoda uusi kilpailu ja hallita olemassa olevaa kilpailua
      - Lomake, johon syötetään kilpailun tiedot
        - Mahdollisuus asettaa muita kilpailun admineiksi
        - Mahdollisuus asettaa kilpailulle useita kierroksia
        - Mahdollisuus asettaa kilpailulle oma pelimuoto
        - Mahdollisuus asettaa pelaajia erilaisiin ryhmiin ja kategorioihin
        - Mahdollisuus lisätä ja poistaa pelaajia kilpailusta
        - Mahdollisuus asettaa kilpailun julkisuusasetuksia
      - Mahdollisuus lähettää tietoa osallistujille
        - Push-notifikaatiot
        - Alustan sisäiset viestit
      - Painike jolla julkaista kilpailu
  - Frontendin koodissa rajapintayhteys backendiin

- Karttanäkymä
  - Näkymä, jossa seuraavanlaiset elementit ja ominaisuudet:
    - Kartta
      - Ominaisuus, jolla näytetään käyttäjän sijaintitietojen perusteella lähinnä olevat radat (haetaan tietokannasta)
      - Ominaisuus, joka piirtää ratojen kartat kartalle (haetaan tietokannasta)
      - Ominaisuus, jolla piirretään käyttäjän radalla käyttämä reitti kartalle sekä tallennetaan tämä tietokantaan
      - Ominaisuus, jolla voidaan merkitä käyttäjän heittopaikka kartalle sekä tallennetaan tämä tietokantaan
  - Ominaisuus, jolla sijaintietojen perusteella tallennetaan peleissä tapahtuvia tapahtumia
  - Ominaisuus, jolla sijaintietojen perusteella voidaan laskea heiton pituus sekä tallennetaan tämä tietokantaan
  - Toimiakseen täysin, vaatii käyttäjän suostumuksen sijaintitietojen käyttöön
  - Frontendin koodissa rajapintayhteys backendiin

3.3 Moduulisuunnittelu

Tässä osiossa käydään yksityiskohtaisemmin läpi eri komponenttien toimintaa kirjallisesti. Tämän osion alta löytyvä tieto myös määrittelee sen, millaiseksi komponentit tulee rakentaa ohjelmointivaiheessa, jotta toiminta olisi vastaavaa kuin suunniteltu. Tämä osa on jaettu kahteen pääosaan, backendiin ja frontendiin, jotka sisältävät oman alueensa komponentit. Koska kyse on fullstack-sovelluksesta, joitakin päällekkäisyyksiä ja riipuvuuksia frontendin ja backendin kesken saattaa ilmaantua. Nämä tulee ottaa huomioon ohjelmointivaiheessa.

1. Backend
   - Käyttäjänhallinta:

     Käyttäjänhallinnaksi kutsutaan käyttäjien tietoja käsitteleviä ominaisuuksia. Esimerkiksi rekisteröityminen ja kirjautuminen ovat osa käyttäjänhallintaa. Käyttäjien frontendissä syöttämä data tuodaan backendille, joka pyyntöjen perusteella määrittelee, miten kyseisessä tilanteessa saatua dataa käsitellään. Backend täten viestii rajapinnan kautta käyttäjienhallintaan liittyvän tietokannan kanssa, hakien, päivittäen tai poistaen tietoa sen mukaan, mitä pyynnössä pyydetään.

Esimerkiksi, käyttäjän rekisteröityessä ja/tai kirjautuessa, otetaan yhteyttä frontendistä rajapinnan kautta backendiin, jossa pyyntö käsitellään. Jos käyttäjää ei ole olemassa, luodaan tietokantaan uusi käyttäjä. Jos käyttäjä on olemassa, varmistetaan käyttäjän tiedot tietokannasta ja näiden ollessa oikein, päästetään käyttäjä kirjautumaan sisään palveluun. Backendissä on syytä noudattaa hyviä tietoturvallisuuden tapoja, eikä frontendiltä tai käyttäjältä saatuun tietoon pidä koskaan implisiittisesti luottaa.

- Tulospalvelun datankäsittely

  Tulospalvelun datankäsittely kattaa kaiken alustan tulospalveluihin liittyvän informaationhallinnan. Backendissä suurin komponentti on tietokanta, johon tulospalvelun tietoja tallennetaan. Lisäksi siihen lukeutuvat rajapinnat frontendin ja backendin väliseen liikenteeseen, sekä backendin sisäiseen liikenteeseen. Tarpeiden mukaan, otetaan yhteys rajapinnan kautta tietokantaan, jossa tietoja voidaan noutaa, lisätä, päivittää tai poistaa.

  Esimerkiksi, kun tulospalvelu saa päivityksiä frontendistä, kuten käyttäjän lisäämät tulokset kilpailussa, käsitellään data backendissä. Lisäksi tietokannasta noudetaan tietoja frontendiin, jotta käyttäjien näkymää voidaan päivittää reaaliajassa. Backendissä on syytä noudattaa hyviä tietoturvallisuuden tapoja, eikä frontendiltä tai käyttäjältä saatuun tietoon pidä koskaan implisiittisesti luottaa.

- Karttapalvelun datankäsittely

  Karttapalvelun datankäsittely kattaa kaiken datan, jota alusta hyödyntää frontendin karttaominaisuuksien pohjana. Käytännännössä backendin puolella tämä on tietokanta, johon voidaan sopivassa tekstimuodossa tallentaa dataa, joka frontendin puolella parsittuna saadaan näkymään ihmisystävällisessä visuaalisessa muodossa. Karttapalvelu sisältää siis karttojen ominaisuuksia, kuten koordinaatteja ja kartan päälle piirrettäviä kuvia esimerkiksi XML-muodossa. Näitä tietoja voidaan noutaa, lisätä, päivittää tai poistaa tietokannasta.

  Esimerkiksi, kun käyttäjä avaa tietyn ratakohtaisen kartan, tekee frontend rajapinnan kautta pyynnön backendiin, joka käsittelee pyynnön. Tämän onnistuessa tietokannasta noudetaan kyseistä pyyntö vastaavat tiedot, jotka toimitetaan frontendiin. Backendissä

on syytä noudattaa hyviä tietoturvallisuuden tapoja, eikä frontendiltä tai käyttäjältä saatuun tietoon pidä koskaan implisiittisesti luottaa.

- Notifikaatioiden lähettäminen

  Notifikaatioiden lähettäminen koostuu kahdesta osasta, sähköpostinotifikaatioista ja push-notifikaatioista. Käytännössä nämä koostuvat omista moduuleistaan, mutta notifikaatioiden lähetystä voidaan hallinnoida samasta näkymästä.

  Sähköpostinotifikaatioita voidaan lähettää backendistä sähköpostimoduulilla. Esimerkiksi, kun alustan administraattorit haluavat lähettää massasähköpostia kaikille käyttäjille, luovat he sähköpostin HTML-muodossa. Tämä sitten lähetetään rajapinnan kautta sähköpostimoduulille, joka postittaa viestin jokaiselle määritetylle vastaanottajalle. Ominaisuutta voidaan myös hyödyntää vaikkapa käyttäjien rekisteröityessä, varmennussähköpostin muodossa.

  Push-notifikaatiot toimivat oman rajapintansa kautta. Näitä voidaan lähettää esimerkiksi suoraan alustan frontend-sovellukseen, jossa on rekisteröity subscription push-notifikaatioiden vastaanottamista varten. Push-notifikaatioilla tulee olla oma moduulinsa backendissä. Moduulia voidaan hallinnoida rajapinnan kautta.

  Esimerkiksi, kun alustalla olevan kilpailun järjestäjät haluavat massatiedottaa jotakin kaikille kilpailuun osallistuville, voivat he frontendissä lähettää näille viestin. Viesti ohjautuu sisäisesti backendille, jossa se käsitellään. Tämän onnistuessa, lähetetään viesti oikeassa muodossa push-notifikaatio-moduulille, joka julkaisee sen. Jos käyttäjien sovellukset ovat asetettu kuuntelemaan push-notifikaatioita, saavat he laitteeseensa ilmoituksen ja tämän sisällön.

- Ulkoisten palveluiden kanssa viestiminen (PDGA:n rajapinta)

  Tällä hetkellä ulkoisilla palveluilla tarkoitetaan PDGA:n rajapintaa. Käytännössä tämä tarkoittaa sitä, että käyttäjän syöttäessä tietoihinsa PDGA:n pelaajanumeron, noudetaan PDGA:n rajapinnan kautta heidän tietokannastaa pelaajan tietoja, jotka tallennetaan käyttäjänhallinnan tietokantaan.

2. Frontend:
    - Navigaatio frontendissä

      Navigaatio frontendissä tarkoittaa käsitteenä sitä pääkomponenttia, jonka kautta käyttäjät liikkuvat sovelluksen sisällä. Navigaatioelementti siis sisältää painikkeet, jotka toimivat linkkeinä sovelluksen eri osiin. Lisäksi navigaatioelementti voi sisältää myös nappulan kirjautuneen käyttäjän uloskirjaukseen.

      Esimerkiksi, kun käyttäjä kirjautuu sisään palveluun ja haluaa siirtyä kilpailun listaukseen, klikkaa tai painaa hän navigaatiomenusta kilpailulistauksen linkkiä. Tällöin sovellus ohjaa käyttäjän oikeaan paikkaan.

      Eri käyttölaitteilla ja/tai näytön skaalauksilla on toisistaan eroavia navigointikomponentteja, mutta niiden sisältö on kaikilla sama.

      Suurella skaalauksella varustettu ikkuna sisältää elementit vierekkäin listattuna, joista päästään navigoimaan sovelluksessa.

      Pienellä skaalauksella varustettu ikkuna sisältää painikkeen, jota painamalla avautuu lista painikkeista, joista päästään navigoimaan sovelluksessa.

      Mobiililaitteilla elementti on karusellimainen palkki, jota voidaan vierittää vasemmalle ja oikealle. Tämä sisältää painikkeet, joita painamalla voidaan navigoida sovelluksessa.

    - Kirjautumis- ja rekisteröitymisnäkymä

      Kirjautumis- ja rekisteröitymisnäkymä sisältää ne käyttäjänhallinnan osat, jotka frontendin puolella hoitavat käyttäjätilien luomisen ja autentikoinnin palveluun. Nämä ovat yhteydessä backendiin, joka omalta osaltaan käsittelee tiedot käyttäjänhallintaan liittyen.

      Frontendissä osaan liittyy kaksi erillistä näkymää; rekisteröitymisnäkymä ja kirjautumisnäkymä.

      Rekisteröitymisnäkymän pääosa koostuu lomakkeesta, johon käyttäjä tunnusta luodessaan syöttää tietonsa. Lomake kysyy tiedot käyttäjänimestä, sähköpostiosoitteesta

ja salasanasta sekä salasanan varmistuksesta. Tiedot voidaan syöttää järjestelmään, joka rajapintayhteyden kautta varmistaa backendistä onnistuuko käyttäjätunnuksen luominen. Jos tämä onnistuu, ohjataan käyttäjä kirjautumisnäkymään. Mahdollisista virheistä rekisteröitymisessä tulee ilmoittaa käyttäjälle sopivalla tavalla.

Kirjautumisnäkymän pääosa koostuu lomakkeesta, johon käyttäjä syöttää kirjautumistietonsa, käyttäjänimen tai sähköpostiosoitteen sekä salasanan. Lisäksi kirjautumissivulta löytyy painike unohtuneen salasanan nollaukseen. Käyttäjän syöttäessä tiedot, lähetetään ne rajapintayhteyden kautta backendille. Jos käyttäjän autentikointi backendissä onnistuu, kirjataan käyttäjä sisään alustalle. Mahdollisista virheistä kirjautumisessa tulee ilmoittaa käyttäjälle sopivalla tavalla.

Käyttäjän tietoja käsitellessä tulee muistaa hyvät tieto- ja kyberturvan periaatteet, kuten salatun yhteyden käyttäminen. Lisäksi mihinkään käyttäjän syöttämiin, tai frontendista saatuihin tietoihin, ei pidä implisiittisesti luottaa ilman filtteröintiä ja verifiointia.

- Profiilinäkymä

  Profiilinäkymällä tarkoitetaan käyttäjien profiilin tiedot sisältävää ikkunaa. Tämä ikkuna on universaali kaikille käyttäjille, mutta sisältö vaihtelee käyttäjän mukaan. Rakenteeltaan profiilin tiedot ovat rakennettu lomakkeena, joka on tyylitelty sopivan näköiseksi ja muuttumattomaksi. Profiilin tietoja varten frontendistä on rajapintayhteys backendiin.

  Profiilinäkymä koostuu lähtökohtaisesti monesta elementistä; käyttäjänimestä, profiilikuvasta, pelaajan peleissä käyttämistä kiekoista ja pelaajan pelihistorian statistiikkaelementistä.

  Lisäksi käyttäjän salliessa, profiilisivulla näytetään lisätietoja; pelaajan oikea nimi, pelaajan kotimaan lippu sekä kotimaan lyhenne ja valinnaisesti pelaajan PDGA-numero.

  Käyttäjän ollessa omalla profiilisivullaan, näytetään lisäksi painike jota painaessa lomakkeesta koostuva profiilisivu muuttuu muokattavaksi. Käyttäjä voi siis muokata tietoja suoraan profiilisivun kentissä, jotka tallennettaessa päivitetään backendin

tietokantaan. Lisäksi tästä näkymästä löytyvät mahdollisuudet muokata käyttäjän profiilin yksityisyysasetuksia.

Käyttäjän tietoja käsitellessä tulee muistaa hyvät tieto- ja kyberturvan periaatteet, kuten salatun yhteyden käyttäminen. Lisäksi mihinkään käyttäjän syöttämiin, tai frontendista saatuihin tietoihin, ei pidä implisiittisesti luottaa ilman filtteröintiä ja verifiointia.

- Ratojen listaus

  Ratojen listaus on näkymä, jossa näytetään käyttäjään nähden karkean etäisyyden mukaan listaus frisbeegolf-radoista. Näkymä koostuu siis listamaisesta pääelementistä, jonka alielementteinä ovat radat listattuna. Tässä osassa näkymää näytetään radan nimi/sijainti, sekä käyttäjän etäisyys kyseiseen rataan.

  Jos käyttäjä valitsee tästä listauksesta jonkin tietyn radan, avautuu radan alle alasvetovalikon kaltainen lisäkenttä, jossa näytetään tarkempia tietoja radasta. Tässä näkymässä näkyvät samat tiedot mitä ylemmässä näkymässä, sekä radan kuvaus, väylien määrä, radan vaikeustaso sekä painikeet kyseisen radan täyteen informaationäkymään ja ajo-ohjeisiin, jotka avautuvat ulkoiseen karttasovellukseen, kuten Google Mapsiin.

  Radan täydessä informaationäkymässä näkyvät radan kaikki tiedot. Näitä tietoja ovat kaikki edellämainitut tiedot, sekä esimerkiksi radan kartta, kaikki radan lähiajan menneet kilpailut, tulevat kilpailut ja mahdolliset yhteystiedot radan kunnossapitoon. Lisäksi tästä näkymästä pääsee luomaan oman kilpailun kyseiselle radalle.

  Frontend viestii rajapintayhteyden kautta backendiin, jonka tietokannasta ratojen tiedot haetaan. Myös kilpailujen ja ratojen välinen yhteys päivitetään backendissä. Rajapintayhteydessä tulee noudattaa hyviä tieto- ja kyberturvatapoja, eikä mihinkään frontendista saatuun dataan tule implisiittisesti luottaa.

- Tulosnäkymä

  Tulosnäkymä on turnaus- ja kilpailukohtainen näkymä, jossa näytetään koostetusti tietoja turnauksen tuloksista. Parhaillaan käynnissä oleville kilpailuille näkymä päivittyy automaattisesti reaaliajassa.

Näkymän pääelementti on listaus, jossa näytetään kaikki kilpailuun osallistuvat kilpailijat, sekä heidän tuloksensa. Lisäksi kilpailun parhaat kolme kilpailijaa näytetään korostettuna. Jos käyttäjä on itse osallistuja kilpailussa, näytetään hänet listauksessa korostettuna muiden pelaajien joukossa.

Kun käyttäjä valitsee listauksesta tietyn kilpailijan, näytetään lisätietoja kyseisestä kilpailijasta. Näitä tietoja ovat esimerkiksi väyläkohtaiset tulokset ja kilpailijan kierroskohtainen statistiikka.

Frontendistä on rajapintayhteys backendiin, jonka tietokannasta tiedot haetaan. Lisäksi backend hoitaa tulosnäkymän ja muiden komponenttien välisten tietojen päivityksen. Rajapintayhteydessä tulee noudattaa hyviä tieto- ja kyberturvatapoja, eikä mihinkään frontendista saatuun dataan tule implisiittisesti luottaa.

- Kilpailunhallinta

  Kilpailunhallinnalla tarkoitetaan näkymää, jossa käyttäjät voivat luoda ja hallinnoida omia turnauksia ja kilpailujaan. Tämä on olennaisesti yhteydessä ratojen listaukseen ja tulosnäkymään.

  Ensinäkymä kilpailunhallinnassa on listaus jo olemassa olevista kilpailuista. Jokaisella näistä kilpailuista on oma infonäkymänsä, jossa näytetään koostetusti kilpailun tietoja. Käyttäjä voi myös liittyä kilpailuun tästä näkymästä. Lisäksi, jos käyttäjä on kilpailun hallinnoija, voi tämä tästä näkymästä siirtyä kilpailun hallintanäkymään.

  Käyttäjän on myös mahdollista luoda uusi kilpailu tästä näkymästä. Käyttäjän luodessa kilpailun, tulee näkyviin lomake johon käyttäjä voi syöttää haluamansa tiedot. Näitä tietoja ovat esimerkiksi; kilpailun nimi ja sijainti, kilpailun hallinnoijat, pelattavat kierrokset, radalla käytettävät väylät, pelimuoto ja että pelaavatko pelaajat erilaisissa ryhmissä tai kategorioissa. Kilpailun voi myös asettaa julkiseksi tai esimerkiksi salasanalla suojatuksi.

  Kilpailun hallinnoijilla on myös mahdollisuus muuttaa kilpailun tietoja ja dataa jälkikäteen. Lisäksi he voivat lähettää viestejä kilpailun osallistujille, jotka näkyvät muille käyttäjille esimerkiksi push-notifikaatioina.

Kilpailujen tiedot tallennetaan frontendin rajapintayhteyden kautta backendin tietokantoihin. Backend myös yhdistää eri komponenttien, kuten ratojen listauksen, tulospalvelun ja kilpailunhallinnan, keskinäisen tiedonvaihdon. Rajapintayhteydessä tulee noudattaa hyviä tieto- ja kyberturvatapoja, eikä mihinkään frontendista saatuun dataan tule implisiittisesti luottaa.

- Karttanäkymä

  Karttanäkymällä tarkoitetaan sovelluksen ominaisuutta, jossa kustomoitu kartta saadaan näkyviin ja käytettäväksi eri puolilla sovellusta. Kartta tulee toimimaan monessa eri komponentissa ja sen toiminta mukautetaan näiden tarkoituksen mukaan.

  Ratojen listauksen näkymässä, karttaominaisuutta voidaan hyödyntää näyttämällä käyttäjän sijaintiin perustuen lähimmät radat. Kartalle voidaan piirtää backendin tietokannasta perustuvan datan perusteella merkinnät radoista.

  Kartalle voidaan myös piirtää ratojen omat ja tarkemmat kartat. Tätä voi hyödyntää esimerkiksi kilpailuissa, jossa halutaan helpottaa fyysistä liikkumista radalla. Pelaajat voivat myös nähdä oman sijaintinsa kartalla, jos he sallivat tarkkojen sijaintitietojen käytön.

  Pelaajan sijaintitietojen perusteella voidaan myös piirtää pelaajan käyttämä reitti radalla. Tämä voidaan tallentaa backendin tietokantaan ja sitä voidaan kilpailujen aikana hyödyntää, näyttämällä kaikkien pelaajien käyttämä reitti koontinäkymässä.

  Lisäksi pelaaja voi merkitä kulloisenkin heittonsa sijainnin kartalle. Tämä tallennetaan backendin tietokantaan, jolloin on mahdollista laskea esimerkiksi heittojen pituuksia. Näitä tietoja voidaan myös hyödyntää kilpailujen koontinäkymissä.

  Moni ominaisuus vaatii tarkkojen sijaintitietojen käytön, joten sovellusta käyttöönotettaessa tulee käyttäjältä kysyä lupaa tarkkojen sijaintitietojen käyttöön.

  Karttanäkymän tiedot haetaan ja tallennetaan frontendin rajapintayhteyden kautta backendin tietokantoihin. Backend jakaa myös muille komponenteille tarvittavia karttaominaisuuden tietoja. Rajapintayhteydessä tulee noudattaa hyviä tieto- ja

kyberturvatapoja, eikä mihinkään frontendista saatuun dataan tule implisiittisesti luottaa.

3.4 Rajapinnat

Suunniteltu ohjelmistokokonaisuus sisältää rajapintoja monenlaiseen eri tarkoitukseen. Nämä voidaan jakaa osittain frontendin ja backendin omiin rajapintoihin, mutta näiden kahden yhdistämiseen on myös omat rajapintansa.

1. Backend

   Backendin sisäiset rajapinnat liittyvät pääasiassa tietokantojen kanssa kommunikoimiseen. Backend kokonaisuutena sisältää monta eri tietokantaa, jotka ovat toimivat kuitenkin omina komponentteinaan. Näiden kanssa kommunikoimiseen täytyy backendissä olla rajapinnat, joihin on rakennettu mallit, joiden mukaan tietoliikennettä käsitellään.

   Rajapinnat siis päättävät saamansa pyynnön mukaan sen, miten tietoa käsitellään, filtteröidään, mihin se ohjataan ja niin edelleen. Ennen pyyntöjen käsittelyä varmistetaan käyttäjiltä saaman datan tietoturvallisuus käsittelyä varten. Myös mahdolliset virhetilanteet raportoidaan rajapintojen toimesta backendille, joka välittää tiedot tarpeen mukaan asiaan liittyville osapuolille.

   Backendissä on myös rajapintoja, joilla hoidetaan kommunikointia ohjelmiston ulkopuolisiin palveluihin. Näitä ovat PDGA:n palvelu, josta noudetaan käyttäjien tietoja, sekä notifikaatioiden lähettäminen sähköpostien ja push-notifikaatioiden muodossa.

   Lisäksi backend tarvitsee rajapintayhteyden frontendin kanssa kommunikoimiseen. Tästä iso osa toimii niin, että frontendin puolella on erilaisia pyyntöjä, jotka se tekee backendille. Backend toimittaa frontendille dataa näiden rajapintojen lävitse. Lisäksi tarvittaessa dataa voidaan päivittää tai poistaa backendissä, pyynnöstä riippuen.

2. Frontend

   Frontendin rajapinnat koostuvat pääasiassa kommunikaatioyhteyksistä alustan backendiin. Näillä rajapintayhteyksillä tehdään siis pyyntöjä, joiden perusteella haetaan, muutetaan tai poistetaan dataa backendistä.

   Käytännössä jokainen frontendin näkymä saa tarvitsemansa informaation backendiltä. Frontendiin tulee siis luoda yhteydet, joilla tehdään pyynnöt tarvittavaan datankäsittelyyn. Esimerkiksi käyttäjien profiilinäkymien, tulosnäkymän ja karttapalvelun data saadaan backendistä omilla pyynnöillään. Myös frontendissä tapahtuneita tietojen muutoksia lähetetään backendiin, omilla pyynnöillään.

   Kaikki nämä pyynnöt käsitellään ja muodostetaan frontendin omissa moduuleissa. Pyynnöt lähetetään tämän jälkeen backendiin, joka puolestaan käsittelee pyynnöt ja ohjaa ne eteenpäin oikeille moduuleille.

   Koska käyttäjältä ja frontendistä saatu data on potentiaalisesti aina vaarallista, ei frontendin moduuleissa suoriteta kosmeettista suurempaa filtteröintiä. Kaikki pyynnöt suodatetaan erikseen backendissä, ennen niiden jatkokäsittelyä. Mahdollisista virhetilanteista ilmoitetaan käyttäjälle asianmukaisilla virheilmoituksilla.

3.5 Riippuvuudet

Ohjelmistokokonaisuudessa on monia riippuvuuksia, vaikka monen komponentin perustoiminta on itsenäistä. Suurin osa näistä riippuvuuksista on ohjelmistokokonaisuuden sisäisiä, mutta myös ulkoisista palveluista ja niiden saatavuudesta ja toiminnasta ollaan riippuvaisia.

Koko ohjelmistokokonaisuus on riippuvainen niistä alustoista ja palveluista, joissa ohjelmistoa pyöritetään. Lisäksi tietokannat ovat vastaavalla tavalla riippuvaisia näistä alustoista. Ohjelmiston pääalustaksi on valittu PaaS-palvelu, joka siis tarjoaa pilvipohtaisen alustan ohjelmiston frontendin, backendin ja näihin liittyvien toimintojen tarjoamiseen. Jos kyseinen palvelu syystä tai toisesta lakkaa toimimasta, on koko ohjelmisto poissa käytöstä. Poikkeuksena tässä on mobiilisovelluksen muutamat perusominaisuudet, jotka toimivat edelleen, joskin rajoitetusti. PaaS-palvelun käyttämisessä on kuitenkin etuna se, että todennäköisyys täydelle käyttökatkolle on todella pieni. Ohjelmiston eri osat

toimivatkin palvelinpuolella luotetun toimittajan virtuaalisissa ympäristöissä, jolloin esimerkiksi käyttökatkoja laiterikkojen vuoksi ei synny.

Backendin puolella sisäisiä riippuvuuksia on monia. Näistä suurin osa on tietokantoja ja näiden rajapintayhteyksiä, joihin ohjelmiston toiminta perustuu. Tietokannat tuleekin hajauttaa omiin palveluihinsa, jotta yhden vikaantuminen ei vaikuta muihin. Tietokannoista kannattaa myös pitää tarvittaessa useampaa reaaliaikaista kopiota sekä varmuuskopiota, jotta virhetilanteessa tietoja ei menetetä, tai jotta tietojen menetys pystytään minimoimaan.

Tietokantojen lisäksi vähäisempiä riippuvuuksia backendissä ovat sovelluksen ulkopuolen kanssa viestivät palvelut, kuten PDGA:n rajapinnat sekä notifikaatioiden lähettäminen palvelun ulkopuolelle. Näiden toimintaongelmat eivät kuitenkaan estä suurinta osaa ohjelmiston käytöstä, mutta voivat aiheuttaa käyttäjille harmillisia kokemuksia.

Frontendin puolella suurin osa riippuvuuksista on yhteydessä backendiin. Frontendin peruskomponentit toimivat itsenäisesti, mutta ilman yhteyttä backendiin ja tämän tietokantoihin, eivät nämä komponentit voi näyttää käyttäjälle tietoja. Esimerkiksi käyttäjien profiilien tietoja ei voida ladata näytettäväksi frontendiin, jos käyttäjienhallinnan tietokantaa tai tämän rajapintayhteyttä ei ole saatavilla. Vastaavanlaisesti tulosnäkymä ei voi näyttää tuoreimpia tuloksia, jos yhteyttä tietokantaan ei ole. Sama periaate toimii pääasiassa kaikilla komponenteilla. Tätä voidaan alustavasti lieventää tallentamalla tietoja käyttäjän laitteen välimuistiin, mutta tämä on kuitenkin rajoitettua. Välimuistiin tallennettu tieto myös tehostaa komponenttien välistä tiedonjakamista, esimerkiksi kilpailunhallinnan ja tulosnäkymän välillä.

Näiden tietokantariippuvuuksien lisäksi, frontend on osittain riippuvainen käyttäjän geolokaatiodatasta. Geolokaatiodata ei ole kuitenkaan välttämätön, sillä käyttäjä voi halutessaan itse estää tämän käytön kokonaan. Jotkin sovelluksen osat, kuten karttapalvelu, eivät kuitenkaan toimi kunnolla jos geolokaatiodataa ei ole saatavilla.

3.6 Saavutettavuus

Ohjelmistokokonaisuus koostuu kahdesta erillisestä käyttöliittymästä, joissa molemmissa pyritään huomioimaan saavutettavuus mahdollisimman hyvin. Web-käyttöliittymä ja mobiilisovellus ovat

toiminnaltaan hyvin samankaltaisia, joten samat periaatteet saavutettavuuden suhteen toimivat molemmissa. Saavutettavuuden kehittämisessä täytyy kuitenkin olla realistinen, kokonaisuus huomioon ottaen. Kaikkia saavutettavuustoimia ei ole mahdollista toteuttaa ilman, että sovelluksen toimintalogiikka ja yleinen käytettävyys ei kärsi. Suunnittelussa on kuitenkin pyritty huomioimaan Web Content Accessibility Guidelines (WCAG) 2.2 mukaisia tärkeitä saavutettavuustoimia.

1. Responsiivinen ja skaalautuva käyttöliittymä

   Käyttöliittymä tulee rakentaa niin, että se toimii mahdollisimman hyvin kaikilla laitteilla sekä kaikilla skaalauksilla. Tämä sisältää sen, että elementit pysyvät selkeinä, ehjinä ja käytettävinä, vaikka sovelluksen skaalaus muuttuisi suhteellisen radikaalisti. Lisäksi sovelluksen navigoinnin tulee pysyä käyttökelpoisena ja esimerkiksi navigointipalkki ja sen toiminta tulee sopeuttaa skaalauksen mukaan. Tätä kutsutaan yleisesti myös responsiivisuudeksi.

2. Kontrastin vaihto

   Sovelluksessa tulee pystyä vaihtamaan yleisnäkymän kontrastia. Näitä tiloja ovat nykypäivänä yleisesti käytetyt light mode ja dark mode, joissa sovelluksen ilme muuttuu tumman ja vaalean tausta välillä. Myös teksti muuttuu väriltään näiden mukaan. Lisäksi sovelluksessa tulee olla mahdollisuus vaihtaa korkean kontrastin tilaan, jossa kaikki elementit ovat selkeästi erottuvia.

3. Fonttikoon vaihtaminen

   Sovelluksessa tulee olla mahdollisuus vaihtaa fonttikokoa käyttäjän tarpeiden mukaan. Tämä voidaan toteuttaa verkkosivuilla yleisesti käytössä olevalla tavalla, jossa on painikkeet fonttikoon suurentamiseen sekä pienentämiseen. Fontiksi voidaan myös valita muutama eri vaihtoehto, joista käyttäjä voi valita itselleen parhaiten sopivan.

4. Kuvaavat vaihtoehtoiset tekstit elementeille

   Jokaisella elementillä, kuten linkeillä ja kuvilla, tulee olla HTML-kielen mukainen alt-teksti (vaihtoehtoinen teksti). Tällä voidaan taata se, että esimerkiksi kuvan oleellinen sisältö voidaan tuoda esiin myös niille, jotka eivät voi saada selvää kuvista. Tämä myös parantaa sovelluksen käytettävyyttä, jos vaikka verkkoyhteys on huono ja kuvat eivät lataudu.

5. Tärkeiden toimintojen varmistus ennen suoritusta

   Sovellus sisältää esimerkiksi monia lomakkeita, joihin käyttäjä voi syöttää tietoja. Lomaketta lähetettäessä sovelluksen tulee varmistaa käyttäjältä, haluaako tämä varmasti lähettää lomakkeen sisällön. Näin saadaan minimoitua virhepainalluksista johtuvat turhat lomakkeiden lähetykset. Samaa periaatetta voidaan soveltaa myös muihin vastaavanlaisiin toimintoihin.

6. Navigoinnin yksinkertaisuus

   Sovelluksen navigointipainikkeiden määrän tulee pysyä järkevänä ja selkeänä. Lisäksi jokaisesta näkymästä tulee päästä takaisin edelliseen näkymään tai sovelluksen kotinäkymään. Kaikkien sovelluksen pääkomponenttien välillä tulee päästä liikkumaan loogisesti ja helposti. Käyttäjä ei saa siis tuntea olevansa hukassa sovelluksessa, tai päätyä umpikujaan josta ei pääse navigoimaan pois. Lisäksi elementtien tulee olla järkevän kokoisia ja suurinta osaa sovelluksesta tulee voida käyttää yhdellä kädellä, jopa hanskat kädessä.

7. Kielen vaihtaminen

   Sovelluksessa tulee voida vaihtaa kieli, suomen ja englannin kielien välillä. Kehityksen edetessä tulee varata mahdollisuus myös muihin lokalisaatioihin.

8. Kielen oikeaoppisuus

   Kielen tulee olla oikeaoppista ja helposti ymmärrettävää. Tämä tarkoittaa siis sitä, että sovelluksen termistö on oikeaa ja yksinkertaista. Tarpeen tullen vaikeaselkoisimmat kohdat tulee olla selitettynä sovelluksessa, esimerkiksi ohjelaatikon muodossa. Kielen tulee olla kirjakieltä.

9. Virheilmoitukset käyttäjälle

   Käyttäjälle tulee näyttää yksinkertaiset, helposti ymmärrettävät ja huomion kiinnittävät virheilmoitukset. Tämä tulee tapahtua aina, kun sovelluksen toiminnassa tapahtuu jonkin virhe, oli se sitten lähtöisin käyttäjästä tai itse sovelluksesta.

4 TUOTTEENHALLINTA

4.1 Versionhallintasuunnitelma

Ohjelmistokokonaisuus koostuu käytännössä neljästä eri pääkomponentista, joita kehitetään erikseen. Nämä pääkomponentit vaativat omat versionhallintansa, jotta esimerkiksi backendiin voidaan tehdä erikseen muutoksia, ilman että se vaikuttaa frontendin toimintaan. Nämä pääkomponentit ovat seuraavat:

1. Backend
2. Web-frontend
3. Android-sovellus
4. iOS-sovellus

Ohjelmisto on suunniteltu kehitettäväksi ja toimitettavaksi rolling release -tyylisesti. Tähän ratkaisuun päädyttiin, sillä tarkoituksena ei ole myydä ohjelmistoa eri asiakkaille, vaan kehittää sovellusta joka soveltuu yleiseen käyttöön kehittyen jatkuvasti ajan myötä. Tämä myös selkeyttää pienen tiimin työskentelyä, sillä kehityksessä voidaan paremmin keskittyä uusien ominaisuuksien luomiseen.

Versionhallintajärjestelmäksi on valittu Git, jota käytetään yhdessä Githubin kanssa. Github repositoryt ovat myös jaettu useaan osaan. Github valittiin käyttöön sen tunnettuvuuden takia, mutta myös siksi että sitä käytettäessä versionhallintajärjestelmän ylläpidosta ja palvelinten toiminnallisuudesta ei tarvitse vastata itse.

Kehityksen aikana jokaiselle käyttöalustalle sekä backendille on omat repositorynsa. Jokaisen kehitysversion uudet ominaisuudet ja päivitykset kehitetään omiin brancheihinsa. Tällä varmistutaan siitä, että jokaista uutta ominaisuutta tai muutosta voidaan seurata ja tarpeen vaatiessa jäljittää. Ominaisuuden valmistuessa, tehdään laatutarkistukset (kuten linttaus ja testien ajaminen), joiden onnistuessa mergetään branch kehitysversion main-branchiin. Kehitysversioista käytetään omaa versionumerointiaan.

Vastaavanlaisesti tuotantoversioille on omat repositorynsä. Tuotantoversio eroaa kehitysversiosta siinä mielessä, että sitä ei varsinaisesti enää kehitetä. Jos ohjelmistoon rakennetaan uusia ominaisuuksia, tehdään sen hetken tuotantoversion pohjalta uusi kehitysversio, johon ominaisuuksia ruvetaan

rakentamaan. Myös mahdolliset bugikorjaukset tehdään tällä tavalla. Tuotantoversioista käytetään omaa versionumerointiaan, joka on hieman suppeampi kuin kehitysversion.

Jokaisesta versiosta tulee myös tehdä versiokohtainen dokumentaatio. Esimerkiksi uutta ominaisuutta kehitettäessä, tulee dokumentoida ominaisuuden toiminnallisuus, kehitystapa ja se miksi ominaisuus kehitettiin. Nämä dokumentit voidaan tallentaa vastaavanlaisesti omaan repositoryynsä, josta kaikki dokumentaatio on helposti löydettävissä ja päivitettävissä.

Ohjelmistoa kehitettäessä ja tuotantoon ajaessa käytetään kahta eri versionumerointia.

1. Kehityksen aikainen versionumerointi

   Ensimmäinen osa kertoo mille alustalle versio kuuluu. Toinen osa on julkaisuajan vuosi, kolmas kuukausi ja neljäs julkaisun järjestysnumero kyseiselle kuukaudelle. Lisäksi kehityksen aikana vaaditaan tarkempaa tietoa jokaisen ominaisuuden versiosta, joten viimeinen osa on yksilöllinen tunniste jokaiselle git-commitille.

   - Backend: Backend vYYYY.MM.X.XXXX
   - Web-frontend: Web vYYYY.MM.X.XXXX
   - Android-sovellus: Android vYYYY.MM.X.XXXX
   - iOS-sovellus: iOS vYYYY.MM.X.XXXX

2. Tuotannon versionumerointi

   Kun tuotantoon julkaistaan päivityksiä, käytetään näillä käytännössä samaa versionumerointia kuin kehityksen aikana, mutta ilman kehityksen aikaista git-commit tunnistetta.

   - Backend: Backend vYYYY.MM.X
   - Web-frontend: Web vYYYY.MM.X
   - Android-sovellus: Android vYYYY.MM.X
   - iOS-sovellus: iOS vYYYY.MM.X

Versionumerointityyli tukee rolling release -julkaisutyyliä. Tämä on selkeä tapa ilmaista käytössä oleva versio sekä kehittäjille että käyttäjille. Esimerkiksi, kun backendistä on tuotannossa versio Backend v2023.02.2, seuraavalle kuulle tarkoitettua päivitystä aletaan työstämään versionumerolla Backend v2023.03.1.12ah.

5 TESTAUS

5.1 Testausstrategia

Ohjelmistokokonaisuutta varten on luotu testausstrategia, jota noudatetaan kaikkien ohjelmiston pääkomponenttien testausta suunnitellessa. Testausstrategia antaa suuntaviivat yleisille toimintatavoille, joiden mukaan komponenttien testauksen tekninen puoli toteutetaan. Testauksen ensisijaisena sidosryhmänä ovat projektin omistajat, jotka ovat myös projektin kehittäjiä, joten strategia on suunniteltu tätä sidosryhmää silmällä pitäen. Testausstrategia kulkee myös monilta osin käsi kädessä laadunvarmistuksen kanssa.

Testauksessa, ja sitä suunnitellessa, referoidaan tässä dokumentissa aiemmin määriteltyihin toiminnallisiin ja ei-toiminnallisiin vaatimuksiin, arkkitehtuuri- ja moduulisuunnitteluun sekä user story -esimerkkeihin. Tarpeen vaatiessa testausta tulee voida myös laajentaa näiden ulkopuolelle.

Testaamiselle on testausstrategiassa asetettu muutamia pääasiallisia vaatimuksia, joilla pyritään varmistamaan kehitettävän ohjelmiston laatua. Lisäksi näillä vaatimuksilla varmistetaan että kehitettävä ohjelmisto on halutunlainen.

1. Ohjelmistokoodin yhteneväisyys

   Ohjelmistokoodin yhteneväisyyden varmistamiseksi otetaan käyttöön jokaiselle käytössä olevalle ohjelmointikielellä linttaus-säännöt. Linttaustyökalut tulee myös ottaa käyttöön jokaisessa kehitysympäristössä, sekä CI/CD-pipelinessa. Kun jokainen kehittäjä käyttää samoja linttaussääntöjä, tulee koodi käytännössä automaattisesti kirjoitettua yhtenevällä tavalla. Tavoitteena on saada koko lähdekoodi näyttämään siltä, kuin se olisi yhden ihmisen kirjoittamaa. Tämä myös parantaa koodin luettavuutta ja tarvittaessa bugien etsintää.

2. Kehitettävien komponenttien testaus

   Kehityksen aikana komponenteille tulee kehittää yksikkötestejä, joilla varmistutaan siitä että kehitettävän komponentin toiminta vastaa alun perin tarkoitettua. Hyvä tapa on kirjoittaa komponentille testit ennen, kuin varsinainen komponentin rakentaminen alkaa. Yksikkötestejä tulee ajaa paikallisesti kehityksen aikana, sekä ennen mahdollista commitia Gitiin. Tämän lisäksi kaikki yksikkötestit ajetaan läpi CI/CD-pipelinessa.

3. Koko ohjelmistoalustan toiminnallisuuden testaus

   Jotta varmistutaan siitä, että ohjelmiston kaikki komponentit toimivat kokonaisuutena keskenään, tulee ohjelmistolle ajaa End-to-End -testejä (E2E-testi). E2E-testejä ajetaan kehityksen aikana paikallisesti, esimerkiksi kun frontendiin kehitetään uutta moduulia joka viestii backendin ja tämän tietokantojen kanssa. Tämän lisäksi kaikki E2E-testit voidaan ajaa ennen mahdollista commitia Gitiin. Ohjelmistokokonaisuuden kaikki E2E-testit ajetaan myös CI/CD-pipelinessä, ja nämä ovatkin erityisen tärkeitä jotta varmistutaan ohjelmiston oikeasta toiminnallisuudesta.

4. Koodianalyysi

   Lähdekoodin laadun varmistamiseksi käytetään kehityksessä koodianalyysiä. Koodianalyysi voi olla reaaliaikaista ja jopa tekoälyllä suoritettavaa. Koodianalyysillä pyritään varmistamaan, ettei kehitettäviin komponentteihin tai moduuleihin ole jäänyt piileviä bugeja, tietoturvahaavoittuvuuksia tai muita ongelmia. Lisäksi koodianalyysillä voidaan varmistaa jo kirjoittamisen aikana ei-toiminnallisten vaatimusten mukaisia suorituskykyvaatimuksia.

5. Uusien testien luominen

   Ohjelmiston kehityksen aikana tulee luoda uusia testejä jokaiselle uudelle ominaisuudelle. Käytännössä kehittäminen on testivetoista, joten testit tulee ensisijaisesti kirjoittaa ennen uusien ominaisuuksia rakentamista. Tällä tavoin varmistutaan siitä, että ohjelmiston kehitys pysyy oikeilla raiteilla, ja että kehitettävät ominaisuudet ovat tarkoituksenmukaisia.

Jotta kehityksen aikana varmistutaan siitä, että kehitettävä ohjelmisto täyttää testausstrategian mukaiset vaatimukset, tulee testejä ajaa mahdollisimman usein läpi. Kehityksen aikana testejä voidaan ajaa paikallisesti, niin koodin kirjoittamisen aikana, kuin mahdollisen commitin lähestyessä.

Lisäksi pull requestien yhteydessä, ajetaan CI/CD-pipelinen läpi ohjelmiston kaikki testit. Jos nämä testit eivät mene läpi, ei pull requestia voida hyväksyä, ennen kuin kaikki virheet ohjelmistossa on korjattu.

Testit ajetaan läpi CI/CD-pipelinessa myös silloin, kun valmis kehitysversio ollaan saattamassa tuotantoon. Tässä sovelletaan samaa periaatetta, eli jos yksikin testi ei mene läpi, ei ohjelmistoa voida saattaa tuotantopalvelimelle, eikä sille anneta uutta versionumeroa.

5.2 Laadunvarmistus

Rakennettavan ohjelmistokokonaisuuden laadunvarmistus ja -hallinta kulkevat melkolailla käsi kädessä testausstrategian kanssa. Testausstrategiassa keskitytään pääasiassa siihen, että ohjelmistoa kehittäessä laatu pysyy käytännön tasolla hyvänä. Laadukkaan ohjelmistokokonaisuuden saavuttamiseksi on kuitenkin erikseen määritelty muutamia ohjeistuksia.

1. Testausstrategian noudattaminen

   Ohjelmiston kehityksen aikana on erittäin tärkeää noudattaa testausstrategiaa. Tällä pyritään varmistumaan siitä, että ohjelmiston käytännön laatu on mahdollisimman korkeaa ja virheetöntä. Yksityiskohtaisen kuvauksen testausstrategiasta saa edellisestä osasta.

2. Auditointi

   Korkean laadun varmistamiseksi on ohjelmistoa kehitettäessä syytä turvautua auditointiin. Kehityksen aikana auditointia tehdään sisäisesti tiiminä. Tällöin pyrkimyksenä on löytää koodista erilaisia virheitä, jotka ovat jääneet kehityksen aikana piiloon. Ohjelmiston lähestyessä tuotantoa, tai ollessa tuotannossa, voidaan myös turvautua kolmannen osapuolen auditointiin, jotta saadaan mahdollisimman kattava kuva mahdollisista ongelmista.

3. Käyttäjätestaus

   Käyttäjätestaus on olennainen osa kehitettävää ohjelmistoa. Omaksi sidosryhmäkseen määritetyt käyttäjätestaajat ovat niin kutsuttuja beta-testaajia ohjelmistolle. He voivat tarjota käyttäjän näkökulmasta annettavaa palautetta ohjelmiston toiminnasta. Tämän lisäksi kaikille käyttäjille tulee tarjota mahdollisuus lähettää ohjelmistosta palautetta ja esimerkiksi bug-reportteja.

4. Koodin uudelleenkäytettävyys

   Ohjelmistokokonaisuuden eri pääosien sisällä olevat komponentit tulee mahdollisuuksien mukaan rakentaa niin, että niitä voidaan uudelleenkäyttää vastaavanlaisissa käyttökohteissa. Tällä pyritään turvaamaan se, että samaa logiikkaa noudattavaa koodia ei tarvitse kirjoittaa joka kerta uudestaan. Samalla myös koodin laatu kyetään pitämään korkeana. Myös ohjelmiston yhteneväisyys toteutuu näin helpommin.

5. Ei-toiminnallisten vaatimusten täyttäminen

   Ohjelmiston ei-toiminnalliset vaatimukset ovat suunniteltu pitämään käyttäjäkokemus ja saavutettavuus mahdollisimman hyvänä. Ohjelmisto tulee siis rakentaa niin, että ei-toiminnalliset vaatimukset täyttyvät.

6. Helppo ylläpidettävyys

   Ohjelmistoa rakennettaessa ja konfiguroitaessa on syytä kiinnittää erityistä huomiota ylläpidettävyyteen. Tarvittaessa voidaan myös rakentaa omia työkaluja helpottamaan ohjelmiston ylläpitoa. Ylläpitoa helpottaa myös se, että ohjelmistokokonaisuuden palvelinpuoli hoidetaan PaaS-palvelussa.

7. Ohjelmiston käytön loogisuus

   Ohjelmiston käytön loogisuudella tarkoitetaan sitä, että ohjelmiston käytön tulee olla mahdollisimman helppoa ja vaivatonta, niin kokeneille kuin uusille käyttäjille. Ohjelmistokokonaisuus tulee siis suunnitella niin, että kokemus on kaikilla alustoilla mahdollisimman yhteneväinen. Lisäksi ohjelmistossa navigoimisen tulee olla mahdollisimman yksinkertaista.

8. Integriteetti ja tietoturva

   Ohjelmistoa ja sen eri osia kehitettäessä ja tuotantoon saatettaessa tulee kiinnittää erityistä huomiota siihen, että integriteetti ja tietoturva ovat mahdollisimman korkealla. Ohjelmistoon ja sen toimintaan ei siis saa vaikuttaa ulkopuolelta millään haitallisella tavalla. Lisäksi käyttäjien tietojen turvallinen käsittely, sen kaikissa muodoissa, tulee taata mahdollisimman pitkälle. Tietoturvaa varten voidaan myös teettää ulkopuolisia auditointeja.

9. Ohjelmistotuotantosuunnitelman validointi

   Ohjelmistokokonaisuuden kehittyessä eteenpäin, tulee myös ohjelmistotuotantosuunnitelmaa päivittää ajan tasalle. Tasaisin väliajoin tehtävä ohjelmistotuotantosuunnitelman validointi takaa sen, että kehitettävä ohjelmisto on tarkoituksenmukainen ja että kehitettävä ohjelmisto vastaa alkuperäistä visiota mahdollisimman hyvin.

10. CI/CD

Ohjelmiston laadun varmistamiseksi, niin kehityksen kuin tuotantoon saattamisen aikana, tulee turvautua CI/CD-ratkaisuun. Tällä saadaan taattua ohjelmiston virheettömyys kehityksen edetessä. Lisäksi tuotantoon saattaessa ohjelmiston tulee olla käytännössä täysin virheetön, joten CI/CD-ratkaisulla voidaan minimoida testattavissa olevat virheet. Lisätietoa löytää edellä olevasta testausstrategiasta.

Toiminnalliset ja ei-toiminnalliset vaatimukset

Frisbeegolf-tulospalvelu

Tunnisteiden nimitys:
SP – Toiminnallinen vaatimus (suunnittelupalaveri)
ET – Ei-toiminnallinen vaatimus

Toiminnalliset vaatimukset

1. **Käyttäjänhallinta**

TUNNISTE	ALKUPERÄ	KUVAUS	PRIORITEETTI
SP-1/001	Suunnittelupalaveri 4.10.2024	Käyttäjän tulee voida luoda tili palveluun (sisältää käyttäjätunnuksen, salasanan, sähköpostin, käyttäjän lähtömaa, uniikki ID)	Välttämätön
SP-1/002	Suunnittelupalaveri 4.10.2024	Käyttäjän tulee voida kirjautua palveluun rekisteröityessä annetuilla tiedoilla	Välttämätön
SP-1/003	Suunnittelupalaveri 4.10.2024	Käyttäjän tulee voida vaihtaa unohtunut salasana uuteen	Välttämätön
SP-1/004	Suunnittelupalaveri 4.10.2024	Käyttäjällä tulee olla mahdollisuus poistaa tilinsä koska tahansa	Välttämätön
SP-1/005	Suunnittelupalaveri 4.10.2024	Käyttäjälle lähetetään vahvistusviesti tilin luomisesta	Välttämätön
SP-1/006	Suunnittelupalaveri 4.10.2024	Käyttäjällä tulee olla mahdollisuus lisätä PDGA-numero omaan profiiliinsa	Korkea

2. **Sovelluksen yleisnäkymä**

TUNNISTE	ALKUPERÄ	KUVAUS	PRIORITEETTI
SP-2/001	Suunnittelupalaveri 4.10.2024	Sovelluksessa tulee olla rekisteröitymisnäkymä (ks. SP-1/001)	Välttämätön
SP-2/002	Suunnittelupalaveri 4.10.2024	Sovelluksessa tulee olla kirjautumisnäkymä (ks. SP-1/002, SP-1/003)	Välttämätön
SP-2/003	Suunnittelupalaveri 4.10.2024	Sovelluksessa tulee olla navigointipalkki, jossa voi siirtyä eri sivujen välillä	Välttämätön
SP-2/004	Suunnittelupalaveri 4.10.2024	Sovelluksessa tulee olla mahdollisuus käyttäjän uloskirjautumiseen	Välttämätön
SP-2/005	Suunnittelupalaveri 4.10.2024	Sovelluksessa tulee olla näkymä käyttäjän omalle profiilisivulle (ks. SP-4)	Välttämätön
SP-2/006	Suunnittelupalaveri 4.10.2024	Sovelluksessa tulee olla näkymä ratojen listaukseen (ks. SP-5)	Välttämätön
SP-2/007	Suunnittelupalaveri 4.10.2024	Sovelluksessa tulee olla karttanäkymä, jossa voi nähdä lähellä olevat radat ja niiden ominaisuudet (ks. SP-6)	Välttämätön
SP-2/008	Suunnittelupalaveri 4.10.2024	Sovelluksessa tulee olla näkymä, jossa voi hallita turnauksia ja kilpailuja (ks. SP-7)	Välttämätön

3. **Tulospalvelun ominaisuudet**

TUNNISTE	ALKUPERÄ	KUVAUS	PRIORITEETTI
SP-3/001	Suunnittelupalaveri 4.10.2024	Näkymässä tulee olla reaaliaikainen listaus meneillään olevista peleistä (ks. SP-3/002)	Välttämätön
SP-3/002	Suunnittelupalaveri 4.10.2024	Klikkaamalla jotakin peleistä avautuu näkymä, jossa avautuu pelin yhteenvetonäkymä (ks. SP-3/003)	Välttämätön
SP-3/003	Suunnittelupalaveri 4.10.2024	Yhteenvetonäkymästä tulee voida avata näkymä, jossa näytetään pelaajat tulosjärjestyksessä, tulosten päivittyessä reaaliajassa	Välttämätön
SP-3/004	Suunnittelupalaveri 4.10.2024	Tulospalvelunäkymässä tulee olla myös selattava historia peleistä, jotka on pelattu aikaisemmin	Korkea

4. **Käyttäjän profiilisivu**

TUNNISTE	ALKUPERÄ	KUVAUS	PRIORITEETTI
SP-4/001	Suunnittelupalaveri 4.10.2024	Profiilisivulla tulee näkyä käyttäjän käyttäjänimi	Välttämätön
SP-4/002	Suunnittelupalaveri 4.10.2024	Profiilisivulla tulee olla mahdollisuus lisätä ja näyttää käyttäjän profiilikuva	Välttämätön
SP-4/003	Suunnittelupalaveri 4.10.2024	Profiilisivulla tulee näkyä mistä maasta käyttäjä on kotoisin, jos tämä on asetettu profiilissa	Välttämätön
SP-4/004	Suunnittelupalaveri 4.10.2024	Käyttäjälle mahdollisuus päättää mitä tietoja omassa profiilissa näytetään julkisesti (yksityisyysasetukset, käyttäjänimi näytetään aina)	Välttämätön
SP-4/005	Suunnittelupalaveri 4.10.2024	Profiilisivulla voidaan näyttää statistiikkaa käyttäjän aikaisemmista peleistä ja/tai turnauksista	Korkea
SP-4/006	Suunnittelupalaveri 4.10.2024	Käyttäjä voi valita näytetäänkö käyttäjän PDGA-numero profiilisivulla (ks. SP-1/006)	Korkea
SP-4/006	Suunnittelupalaveri 4.10.2024	Profiilisivulla tulee olla mahdollisuus näyttää käyttäjän peleissä käyttämät kiekot (in-the-bag -ominaisuus)	Matala

5. **Ratojen listaus**

TUNNISTE	ALKUPERÄ	KUVAUS	PRIORITEETTI
SP-5/001	Suunnittelupalaveri 4.10.2024, päivitetty 13.11.2024	Näkymässä tulee olla listaus radoista, jotka ovat järjestetty etäisyyden mukaan käyttäjään nähden. Näkymässä tulee myös näkyä radan karkea etäisyys (ks. SP-6/002)	Välttämätön
SP-5/002	Suunnittelupalaveri 4.10.2024	Klikkaamalla rataa, näytetään alasvetolaatikossa radan ominaisuuksia (ks. SP-5/003)	Välttämätön
SP-5/003	Suunnittelupalaveri 4.10.2024	Alasvetolaatikossa näytetään radan kuvaus, väylien määrä, radan vaikeustaso	Välttämätön
SP-5/004	Suunnittelupalaveri 4.10.2024	Avaamalla radan klikkaamalla tai koskettamalla, siirrytään radan omaan näkymään, jossa näytetään radan kaikki tiedot	Välttämätön

6. **Karttaominaisuus**

TUNNISTE	ALKUPERÄ	KUVAUS	PRIORITEETTI
SP-6/001	Suunnittelupalaveri 7.10.2024	Sovelluksen tulee pystyä hyödyntämään, tarkastelemaan ja tallentamaan mobiililaitteiden sijaintitietoja (jos käyttäjä antaa tälle oikeudet)	Välttämätön
SP-6/002	Suunnittelupalaveri 7.10.2024, asiakaskysely	Sovelluksessa tulee olla karttaominaisuus, jossa näytetään käyttäjän sijaintitietojen perusteella käyttäjää lähinnä olevat radat	Korkea
SP-6/003	Suunnittelupalaveri 7.10.2024	Sovelluksessa tulee olla sisäänrakennettuna ratojen kartat, jotka voidaan piirtää karttaominaisuuden avulla kartan päälle	Kohtalainen
SP-6/004	Suunnittelupalaveri 7.10.2024	Sovelluksen karttaominaisuuden ja sijaintitietojen pohjalta tulee voida tallentaa väylällä tapahtuvia tapahtumia (ks. SP-6/005-SP-6/007)	Kohtalainen
SP-6/005	Suunnittelupalaveri 7.10.2024	Sijaintitietojen perusteella tulee voida tallentaa käyttäjän sijainti tietokantaan, esimerkiksi heittopaikan merkitsemistä varten	Kohtalainen
SP-6/006	Suunnittelupalaveri 7.10.2024, asiakaskysely	Sijaintitietojen perusteella tulee voida mitata heiton pituus, asettamalla heiton paikan ja määrittämällä heiton päämäärä	Kohtalainen
SP-6/007	Suunnittelupalaveri 7.10.2024	Käyttäjän tulee voida määrittää millä radan väylällä hän pelaa, jotta sijaintitietoja hyödyntämällä voidaan piirtää kartalle pelaajan käyttämä reitti radalla	Kohtalainen

7. **Kilpailujen ja turnausten hallinta**

TUNNISTE	ALKUPERÄ	KUVAUS	PRIORITEETTI
SP-7/001	Suunnittelupalaveri 7.10.2024	Alustassa tulee olla mahdollisuus luoda uusi kilpailu tai turnaus	Välttämätön
SP-7/002	Suunnittelupalaveri 7.10.2024	Kilpailuun tai turnaukseen tulee pystyä valita käytettävä rata ja radalla käytettävät väylät	Välttämätön
SP-7/003	Suunnittelupalaveri 7.10.2024	Kilpailun tai turnauksen luojan tulee voida lisätä muita käyttäjiä osallistujiksi	Välttämätön
SP-7/004	Suunnittelupalaveri 7.10.2024	Kilpailulle tai turnaukselle tulee voida lisätä muita käyttäjiä hallinnoijiksi / admineiksi	Välttämätön
SP-7/005	Suunnittelupalaveri 7.10.2024	Kilpailulle tai turnaukselle tulee voida asettaa salasana, jonka perusteella käyttäjät voivat vapaasti liittyä	Välttämätön
SP-7/006	Suunnittelupalaveri 7.10.2024	Kilpailulle tai turnaukselle tulee voida luoda useita kierroksia, joita pelataan järjestyksessä	Välttämätön
SP-7/007	Suunnittelupalaveri 7.10.2024	Kilpailuihin tai turnauksiin tulee voida lisätä luokkia, joiden perusteella voidaan jakaa pelaajia kategorisesti eri ryhmiin, jotka kilpailevat keskenään	Välttämätön
SP-7/008	Suunnittelupalaveri 7.10.2024	Kilpailun tai turnauksen hallinnoijien tulee voida poistaa pelaajia kilpailusta	Välttämätön
SP-7/009	Suunnittelupalaveri 7.10.2024	Kilpailuilla tai turnauksilla tulee olla oma infosivu, jossa järjestäjät voivat määrittää vapaasti tietoja kilpailusta	Välttämätön
SP-7/010	Suunnittelupalaveri 7.10.2024	Jos kilpailun tiedoissa tapahtuu muutoksia, tulee niistä ilmoittaa osallistujille esimerkiksi push-notifikaatiolla, tai vastaavalla menetelmällä	Välttämätön
SP-7/011	Suunnittelupalaveri 7.10.2024	Turnaukselle tai kilpailulle tulee voida määrittää pelimuoto, kuten yksilöpeli, paripeli tai skins	Välttämätön

8. **Tulosten kirjaaminen ja näyttäminen**

TUNNISTE	ALKUPERÄ	KUVAUS	PRIORITEETTI
SP-08/001	Suunnittelupalaveri 7.10.2024	Sovelluksessa tulee olla näkymä, jossa näytetään kilpailun tai turnauksen aikainen, reaaliaikaisesti päivittyvä tuloskortti	Välttämätön
SP-08/002	Suunnittelupalaveri 7.10.2024	Käyttäjien tulee myös itse voida luoda tuloskortti omia henkilökohtaisia pelejään varten	Välttämätön
SP-08/003	Suunnittelupalaveri 7.10.2024	Käyttäjän tulee pystyä kirjaamaan väyläkohtainen tulos tuloskorttiin (sisältää myös rangaistukset)	Välttämätön
SP-08/004	Suunnittelupalaveri 7.10.2024	Kilpailun hallinnoitsijoiden tulee voida luoda yhteinen tuloskortti kaikille ryhmille, jotka osallistuvat kilpailuun	Välttämätön
SP-08/005	Suunnittelupalaveri 7.10.2024	Tulosten kirjaamisen tulee olla mahdollista vain kilpailun hallitsijoiden määrittämän aikaikkunan sisällä (esimerkiksi tii-ajan alusta tuloskortin palauttamiseen asti)	Välttämätön
SP-08/006	Suunnittelupalaveri 7.10.2024	Käyttäjän tulee voida lisätä luomallensa kortille muita käyttäjiä, joiden tietoja on mahdollista lisätä kortille	Välttämätön
SP-08/007	Suunnittelupalaveri 7.10.2024	Kilpailun luojat voivat merkitä käyttäjän kilpailun toimitsijaksi, joka voi itsenäisesti lisätä kaikkien kilpailijoiden tietoja tuloskortille	Kohtalainen
SP-08/008	Suunnittelupalaveri 7.10.2024	Tuloskortin tuloksia tulee pystyä muokkaamaan jälkikäteen turnauksen hallitsijoiden toimesta	Välttämätön

9. **Sekalaiset ominaisuudet**

TUNNISTE	ALKUPERÄ	KUVAUS	PRIORITEETTI
SP-9/001	Suunnittelupalaveri 7.10.2024	Käyttäjällä tulee olla mahdollisuus jakaa turnauksen tuloskortti sosiaaliseen mediaan	Kohtalainen
SP-9/002	Suunnittelupalaveri 7.10.2024	Käyttäjällä tulee olla mahdollisuus aloittaa puttiharjoitus-pelitila	Kohtalainen
SP-9/003	Suunnittelupalaveri 11.10.2024, päivitetty 13.11.2024	Sovelluksessa tulee olla ohje-sivu, jossa on selitettynä sovelluksen toiminnallisuus ja miten ominaisuuksia käytetään käyttäjän näkökulmasta	Korkea
SP-09/004	Suunnittelupalaveri 11.10.2024	Ensimmäisellä käynnistyskerralla tulee näyttää ns. tutoriaali, jossa näytetään miten sovelluksen päätoiminnallisuus toimii	Kohtalainen

Ei-toiminnalliset vaatimukset

TUNNISTE	ALKUPERÄ	KUVAUS	PRIORITEETTI
ET-1/001	Suunnittelupalaveri 11.10.2024	Käyttäjien luottamukselliset tiedot tulee olla salattuja ja turvallisesti tallennettuna tietokantaan	Välttämätön
ET-1/002	Suunnittelupalaveri 11.10.2024	Käyttäjien yksilöiviä tietoja tulee käsitellä paikallisten lakien ja asetusten mukaisesti	Välttämätön
ET-1/003	Suunnittelupalaveri 11.10.2024	Käyttäjän tulee voida muuttaa ja poistaa omia tietojansa koska tahansa	Välttämätön
ET-1/004	Suunnittelupalaveri 11.10.2024	Käyttäjällä tulee olla mahdollisuus saada selville mitä tietoja hänestä on tallennettu järjestelmään	Välttämätön
ET-1/005	Suunnittelupalaveri 11.10.2024	Järjestelmän tulee kestää 1000 samanaikaista käyttäjää ilman merkittäviä vaikutuksia suorituskykyyn	Välttämätön
ET-1/006	Suunnittelupalaveri 11.10.2024	Järjestelmän tulee skaalautua niin, että äkilliset käyttäjäpiikit aiheuttavat minimaalisen vaikutuksen järjestelmän toimintaan	Välttämätön
ET-1/007	Suunnittelupalaveri 11.10.2024	Sovelluksen mobiiliversion tulee avautua alle viidessä sekunnissa kuluttajaluokan laitteistolla 4G-verkossa	Korkea
ET-1/008	Suunnittelupalaveri 11.10.2024	Sovelluksen selainversion tulee avautua alle kolmessa sekunnissa kuluttajaluokan laitteistolla ja 100M latausnopeudella	Korkea
ET-1/009	Suunnittelupalaveri 11.10.2024	Mobiilisovelluksessa tulee olla myös perustoiminnallisuus ilman internetyhteyttä	Välttämätön
ET-1/010	Suunnittelupalaveri 11.10.2024	Järjestelmän tulee olla yksinkertainen ja selkeä käyttää, niin selainversiolla kuin mobiililaitteilla	Korkea

(jatkuu seuraavalla sivulla)

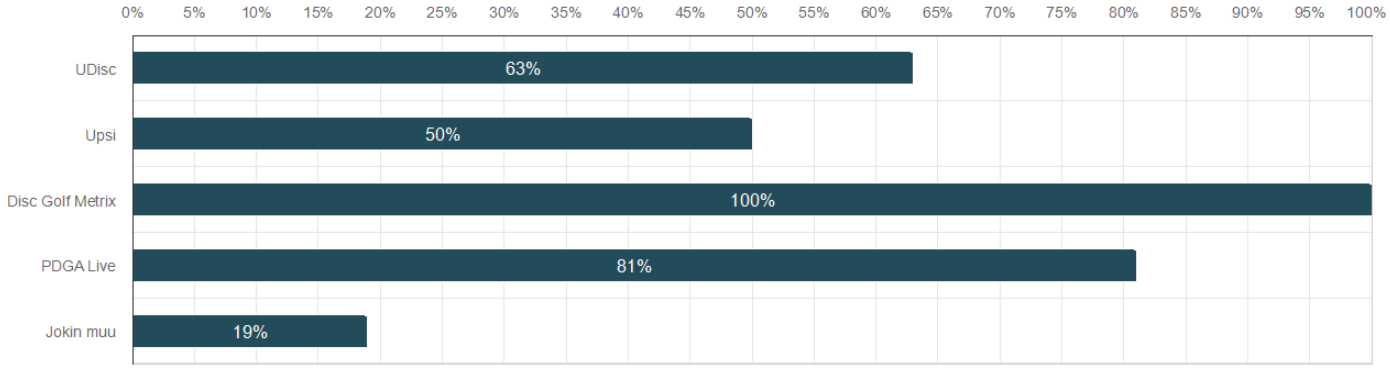
ET-1/011	Suunnittelupalaveri 11.10.2024	Alustan tulee olla saavutettavissa 99% prosenttia liukuvasta, 30 päivän mittaisesta ajasta	Välttämätön

Ohjelmistokehitys
Perusraportti
Frisbeegolf-tulosalusta

Vastaajien kokonaismäärä: 16

Mitä frisbeegolfiin liittyviä palveluja olet käyttänyt?

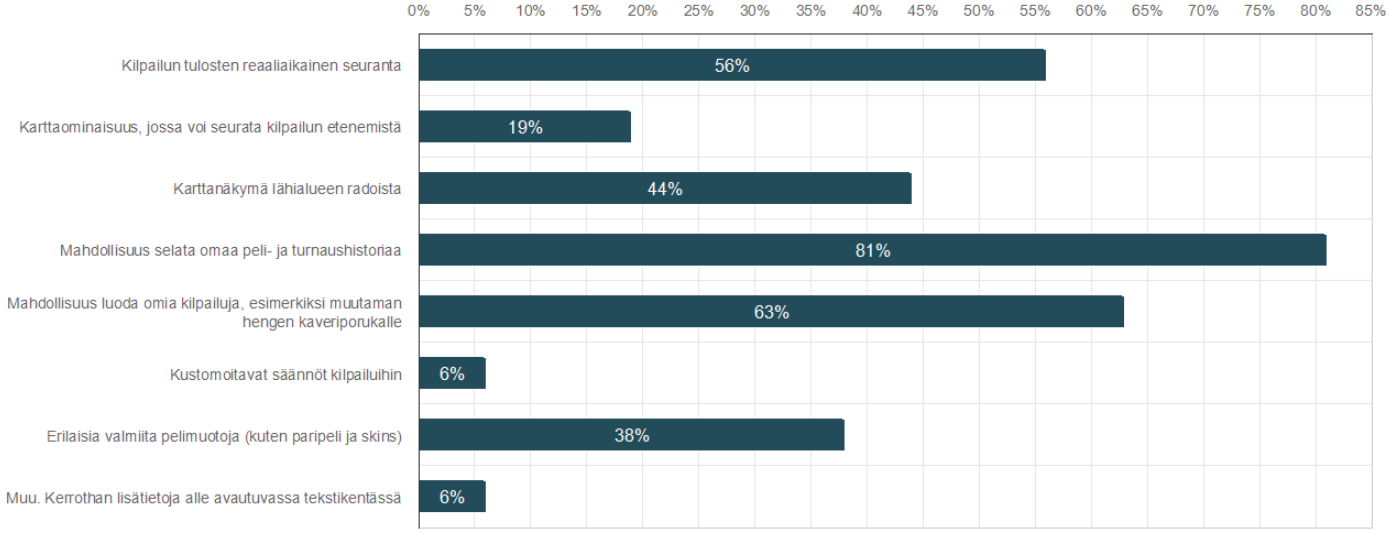
Vastaajien määrä: 16, valittujen vastausten lukumäärä: 50



	n	Prosentti
UDisc	10	62,5%
Upsi	8	50,0%
Disc Golf Metrix	16	100,0%
PDGA Live	13	81,3%
Jokin muu	3	18,8%

Mistä seuraavista ominaisuuksista olet eniten kiinnostunut frisbeegolf-tulosalustoissa?

Vastaajien määrä: 16, valittujen vastausten lukumäärä: 50



	n	Prosentti
Kilpailun tulosten reaaliaikainen seuranta	9	56,3%
Karttaominaisuus, jossa voi seurata kilpailun etenemistä	3	18,8%
Karttanäkymä lähialueen radoista	7	43,8%
Mahdollisuus selata omaa peli- ja turnaushistoriaa	13	81,3%
Mahdollisuus luoda omia kilpailuja, esimerkiksi muutaman hengen kaveriporukalle	10	62,5%
Kustomoitavat säännöt kilpailuihin	1	6,3%
Erilaisia valmiita pelimuotoja (kuten paripeli ja skins)	6	37,5%
Muu. Kerrothan lisätietoja alle avautuvassa tekstikentässä	1	6,3%

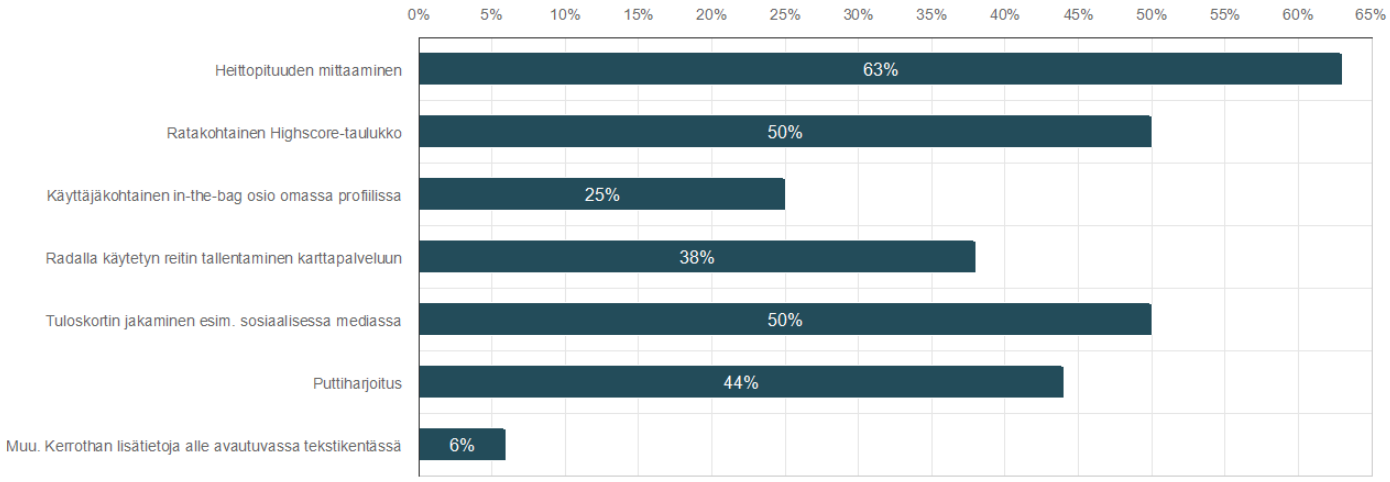
Lisätietoja ominaisuudesta

Vastaajien määrä: 1

Lisätietoja:
Helppokäyttöinen kirjaus

Millaisista lisäominaisuuksista olet kiinnostunut?

Vastaajien määrä: 16, valittujen vastausten lukumäärä: 44



	n	Prosentti
Heittopituuden mittaaminen	10	62,5%
Ratakohtainen Highscore-taulukko	8	50,0%
Käyttäjäkohtainen in-the-bag osio omassa profiilissa	4	25,0%
Radalla käytetyn reitin tallentaminen karttapalveluun	6	37,5%
Tuloskortin jakaminen esim. sosiaalisessa mediassa	8	50,0%
Puttiharjoitus	7	43,8%
Muu. Kerrothan lisätietoja alle avautuvassa tekstikentässä	1	6,3%

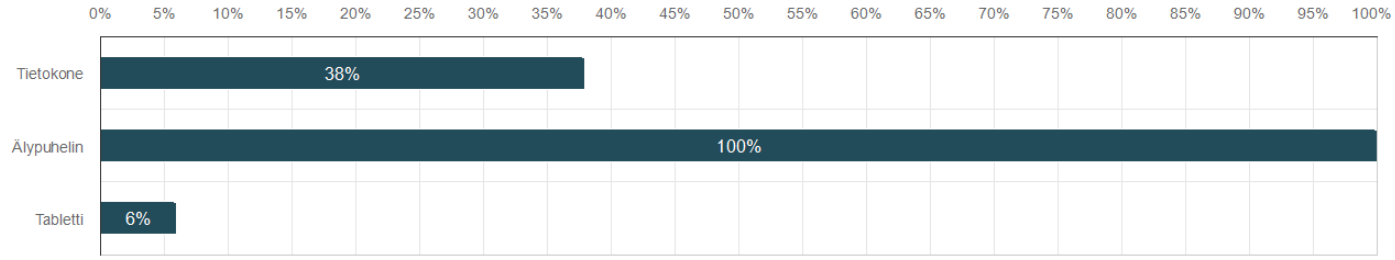
Lisätietoja lisäominaisuuksista

Vastaajien määrä: 1

Lisätietoja
Jonkinlainen ennusteominaisuus perustuen pelaajan aikaisempiin kilpailukierroksiin. Näin näkisi jo kilpailun alusta asti mahdollisen lopputuloksen. Ennuste päivittyisi jatkuvasti kilpailun edetessä. Tämä olisi siis jokin koneoppimiseen pohjautuva sovellus.

Millä seuraavista alustoista olisit kiinnostunut käyttämään sovellusta?

Vastaajien määrä: 16, valittujen vastausten lukumäärä: 23



	n	Prosentti
Tietokone	6	37,5%
Älypuhelin	16	100,0%
Tabletti	1	6,3%

Mitä ominaisuuksia olet jäänyt kaipaamaan olemassa olevissa palveluissa?

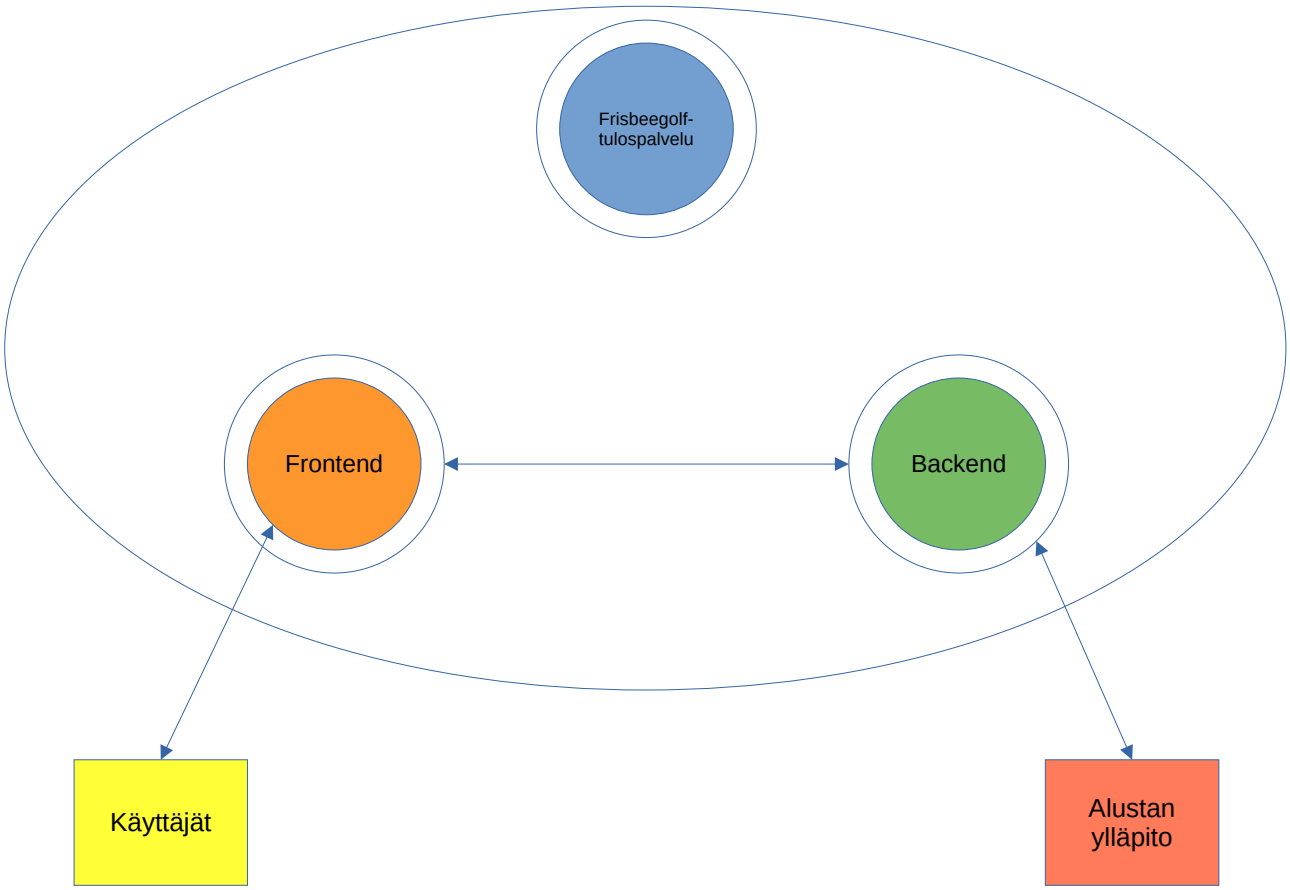
Vastaajien määrä: 3

Vastaukset
Esim metrixissä voisi olla tulostenkirjausnäkymässä sijoitusseuranta, esim. niin, että näkyy omaa sijaa 4 lähintä pelaajaa ja heidän kokonaistulos.
Selkeä ja yksinkertaistettu kisaluettelo jotka saa järjestettyä aika järjestykseen milloin ilmoittautuminen aukeaa. Tai karttapalvelu jossa näät kisat jotka on avoinna ilmottautumiselle.
Karttoja

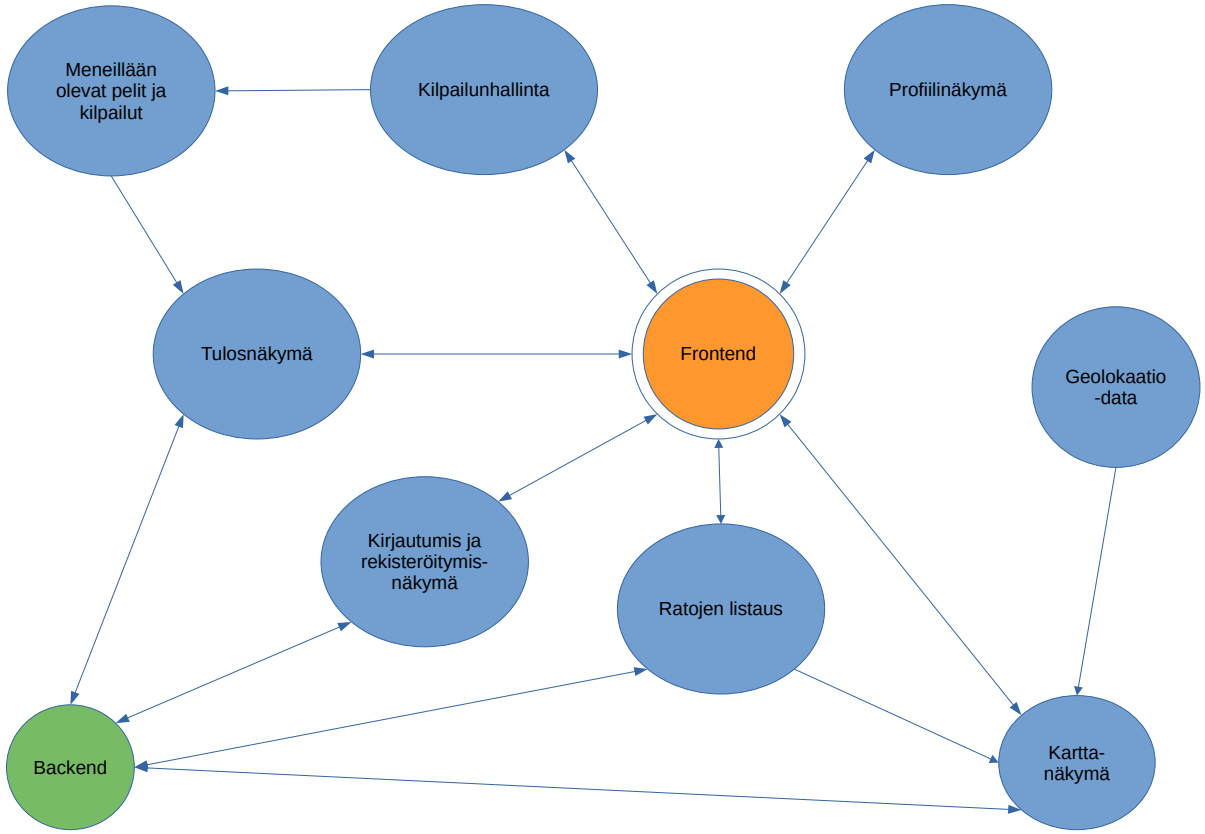
Jos sinulle tulee mieleen jotain muita ideoita, voit kirjoittaa ne vapaasti alle!

Vastaajien määrä: 2

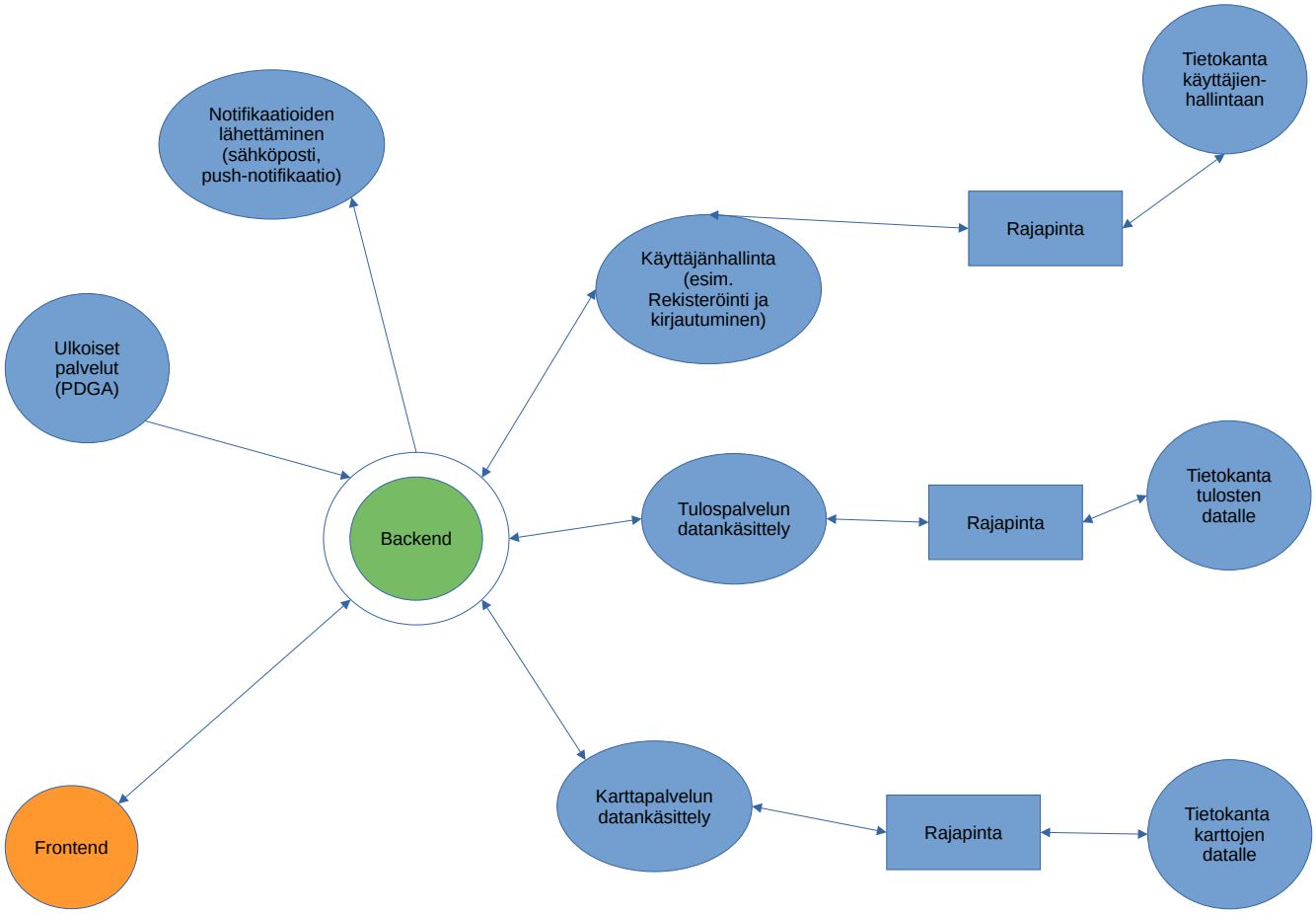
Vastaukset
Selkeä ja yksinkertaistettu kisaluettelo jotka saa järjestettyä aika järjestykseen milloin ilmoittautuminen aukeaa. Tai karttapalvelu jossa näät kisat jotka on avoinna ilmottautumiselle.
Tarkempaa dataa radoista.



Frontend ja backend viestivät toistensa kanssa ja muodostavat kokonaisuuden.



Frontend sisältää eri komponentteja, jotka keskustelevat toistensa kanssa. Lisäksi frontend viestii backendin kanssa.



Backend koostuu useasta tietokannasta, joihin ollaan yhteydessä rajapintojen kautta. Backend voi lisäksi viestiä järjestelmän ulkopuolelle, sekä tietenkin frontendin kanssa.