Sylvester Salo, Julia Köykkä, Jonathan Methuen, Vilhelm Niemi

Speedoku Royale -peliohjelmisto

Metropolia Ammattikorkeakoulu

Insinööri (AMK)

Tieto- ja viestintäteknologian tutkinto-ohjelma

Tekninen dokumentaatio

15.12.2022

Sisällys

1 Johdanto

Dokumentin tavoitteena on täsmentää Speedoku Royale -peliohjelmiston visiota, rakennetta, tavoitteita, tarvetta ja toimintaideaa, sekä niihin liittyviä vaatimuksia. Dokumentissa käydään läpi käytetyt kehitysympärisöt, projektin eri kehitysvaiheet, sekä ohjelmiston rakenne ja toimintatavat eri kaavioiden muodossa.

Dokumentti on suunnattu välittämään tietoa henkilöille, jotka eivät ole olleet projektin kehityksessä mukana. Dokumentin avulla myös kootaan ja päivitetään tietoa projektin kehittäjille pelin jatkokehitystä varten.

Speedoku Royale -pelissä pelaajat kilpailevat toisiaan vastaan sudokujen ratkaisussa. Peli toimii “Battle Royale” -tyylillä. Yhden pelisession alussa pelaajia on tietyn verran ja pelin edetessä pelaajia tippuu, kunnes viimeisenä jäljellä oleva pelaaja voittaa. Speedoku Royale -pelissä pudotetaan tietyn aikavälin välein sudokuista vähiten pisteitä kerännyt pelaaja. Eniten pisteitä kerännyt pelaaja selviää pelin loppuun asti ja voittaa. Ratkaistavat sudokut vaikeutuvat pelin edetessä.

2 Tuotteen vaatimukset

Tarve ja toimintaidea

Ohjelmisto on WebGL -peli, jossa käyttäjät viettävät aikaa peliä pelaamalla. Pelissä ydintavoitteena on ratkaista sudokuja yksin tai reaaliajassa toisia pelaajia vastaan kilpailuhenkisesti. Peli kehitettiin tällä idealla, koska valikoima peleistä, joissa pelataan toisia vastaan sudokuja ratkaisemalla, on puutteellinen.

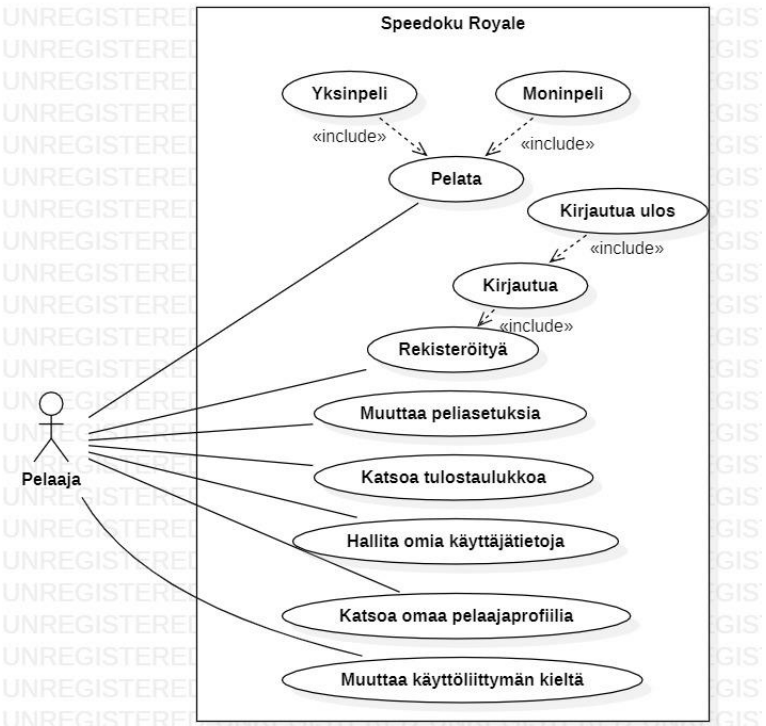
Ohjelmiston vaatimukset

Tavoitteena oli luoda vankka ja toimiva tietokonepeli, jota on vaivaton käyttää ja joka on ulkoasultaan tyylikäs. Toiminnallisia vaatimuksia ohjelmistolle oli toimivat peliympäristöt yksin- ja moninpelille, sekä ymmärrettävät valikot lisäominaisuuksineen, kuten toiminnallisuudet pelattujen pelien tuloksien kirjaamiseen ja niiden myöhempään tarkasteluun.

Moninpelin toimintaidea on antaa pelaajien kilpailla sudokujen ratkaisemisessa. Kyseisessä pelimuodossa sudoku numeroita nopeimmin ratkaiseva pelaaja voittaa. Pelin edetessä ratkaistavien numeroiden määrä kasvaa, jotta sudokujen vaikeusaste nousee.

Peli alkaa, kun pelihuoneeseen on liittynyt kolme pelaajaa ja tietyn ajan kuluttua peli tiputtaa vähiten sudoku numeroita ratkaisseen pelaajan. Aikarajan loputtua viimeiseksi selvinnyt pelaaja julistetaan voittajaksi.

3 Käyttäjäroolit ja käyttötapaukset



Käyttäjäroolit

Pelaaja: Ohjelmiston pääkohderyhmä, jonka tavoitteena on pelata peliä. Pelaaja voi käyttää pelin ominaisuuksia liikkumalla pelin sisäisissä valikoissa.

Käyttötapaukset

Pelaaja voi...

Pelata

- Pelin pelaaminen halutussa pelimuodossa. Käyttäjä voi halutessaan poistua takaisin valikkoon kesken pelin.

Pelata yksin

- Pelaaja ratkoo sudokuja yksin, kilpaillen aikaa vastaan. Pelin loputtua, pelaaja näkee saavuttamansa pistemäärän, jonka jälkeen hän voi aloittaa uuden pelin tai poistua valikkoon.

Pelata muita pelaajia vastaan

- Pelaaja kilpailee reaaliajassa muita pelaajia vastaan. Peli tiputtaa tietyn aikavälin välein pelaajia pelistä. Viimeiseksi selvinnyt pelaaja julistetaan voittajaksi.

- Pelin loppuruudun näkymä määräytyy sen mukaan, voittiko pelaaja pelin vai ei.

- Moninpeli vaatii käyttäjää olemaan kirjautuneena sisälle.

Rekisteröityä

- Uusi käyttäjä rekisteröityy järjestelmään syöttämällä haluamansa käyttäjänimen ja salasanan. Salasana täytyy syöttää täysin samanlaisena kahteen eri kenttään.

- Rekisteröidyttyään käyttäjä kirjataan automaattisesti sisään järjestelmään.

- Jos rekisteröinti ei onnistunut, käyttäjä palautetaan takaisin rekisteröintinäkymään.

Kirjautua

- Rekisteröitynyt käyttäjä syöttää käyttäjänimen ja salasanan.

- Käyttäjätietojen ollessa oikein, käyttäjä kirjataan sisään ja ruutuun tulee näkymä ilmoittaen kirjautumisen onnistumisesta.

- Jos kirjautuminen ei onnistunut, käyttäjä palautetaan kirjautumisnäkymään.

Kirjautua ulos

- Kirjautuneen käyttäjän näkymässä sisäänkirjautumisnappi muutetaan uloskirjautumisnapiksi, jota painamalla käyttäjä kirjataan ulos järjestelmästä.

Muuttaa peliasetuksia

- Käyttäjä voi muuttaa pelin äänien voimakkuutta. (Ominaisuus ei ole vielä toiminnassa)

Katsoa tulostaulukkoa

- Kaikki moninpelimuodossa pelattujen pelien korkeimmat tulokset kirjataan tulostaulukkoon, jota käyttäjä voi halutessaan katsoa.

Hallita omia käyttäjätietoja (Ominaisuus ei ole vielä toiminnassa)

- Kirjautunut käyttäjä voi muuttaa käyttäjätilinsä salasanaa tai poistaa tilinsä pysyvästi

Katsoa omaa pelaajaprofiilia

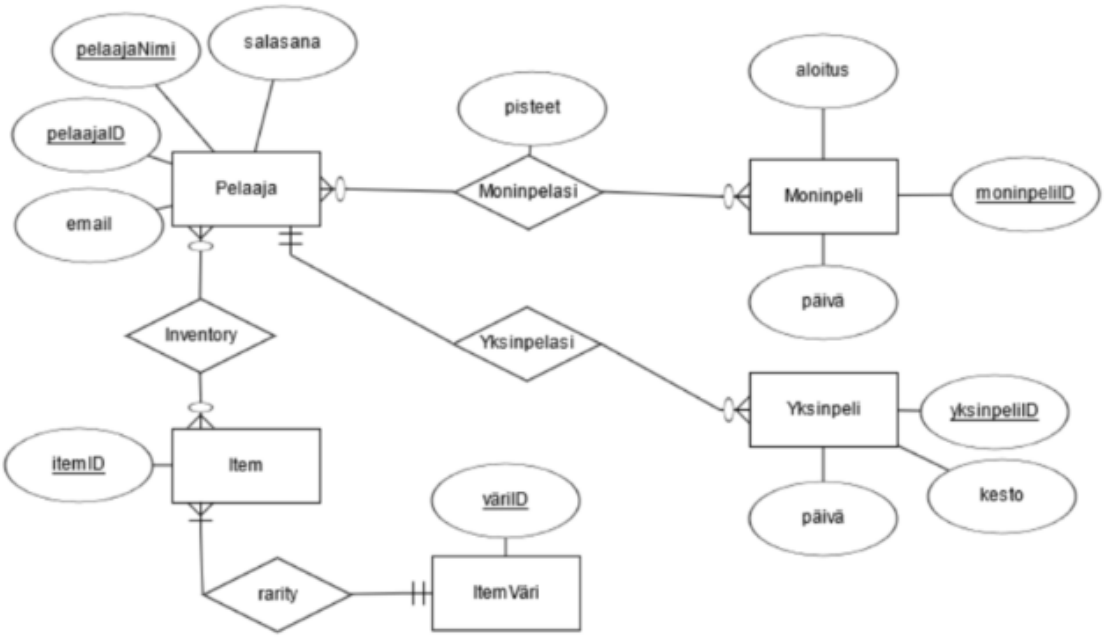
- Kirjautunut käyttäjä voi katsoa omaa profiiliaan, jossa on listattuna hänen moninpelissä voittamien pelien määrä, sekä hänen korkein pelissä saavuttamansa tulos.

Muuttaa käyttöliittymän kieltä

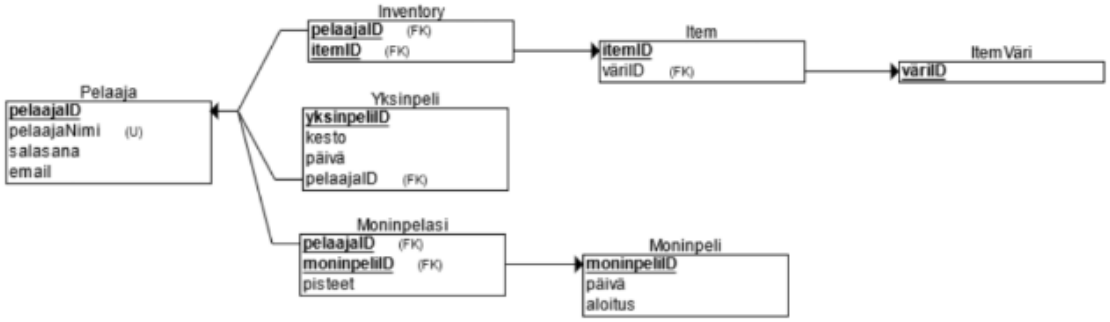
- Käyttäjä voi muuttaa käyttöliittymässä käytettävää kieltä englannin- ja japanin kielten välillä.

4 Ohjelmiston tietomalli

ER-kaavio



Relaatiotietokantakaavio



5 Ohjelmiston rakenne

Ohjelmiston luokkakaaviot eivät mahdu kokonaisuudessaan tarpeeksi isoina kuvina tähän dokumenttiin, joten ne ovat seuraavien linkkien takana.

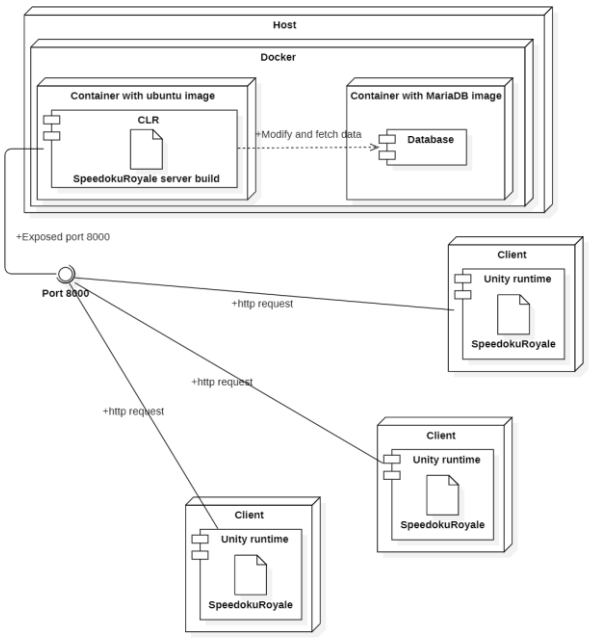
Pelin luokkakaavio:

https://github.com/Jonanananas/SpeedokuRoyale/blob/main/Documents/SpeedokuRoyale_ClassDiagram.pdf

Palvelimen luokkakaavio:

https://drive.google.com/file/d/1zOPhxyNrbU6lGHQ1oBhRn-WliTcmMa1bg/view?usp=sharing

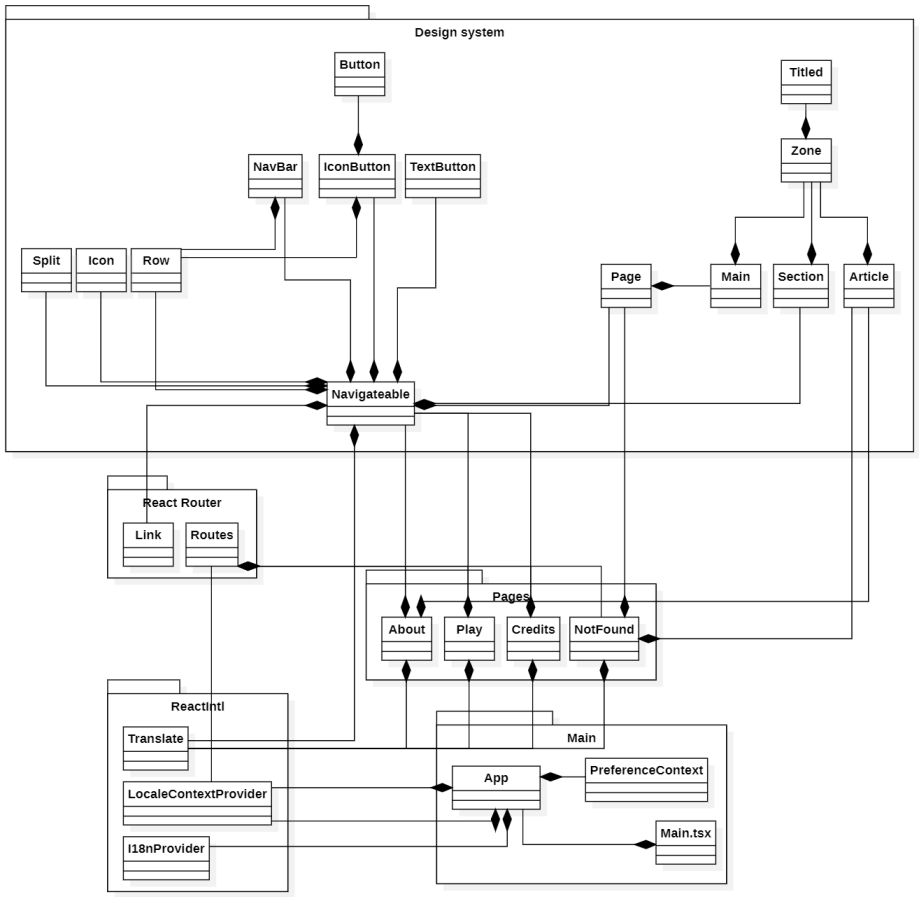
Ohjelmiston fyysinen rakenne tuotantokäytössä.



Luokkakaaviolla demonstroitu verkkosivun komponenttirakenne

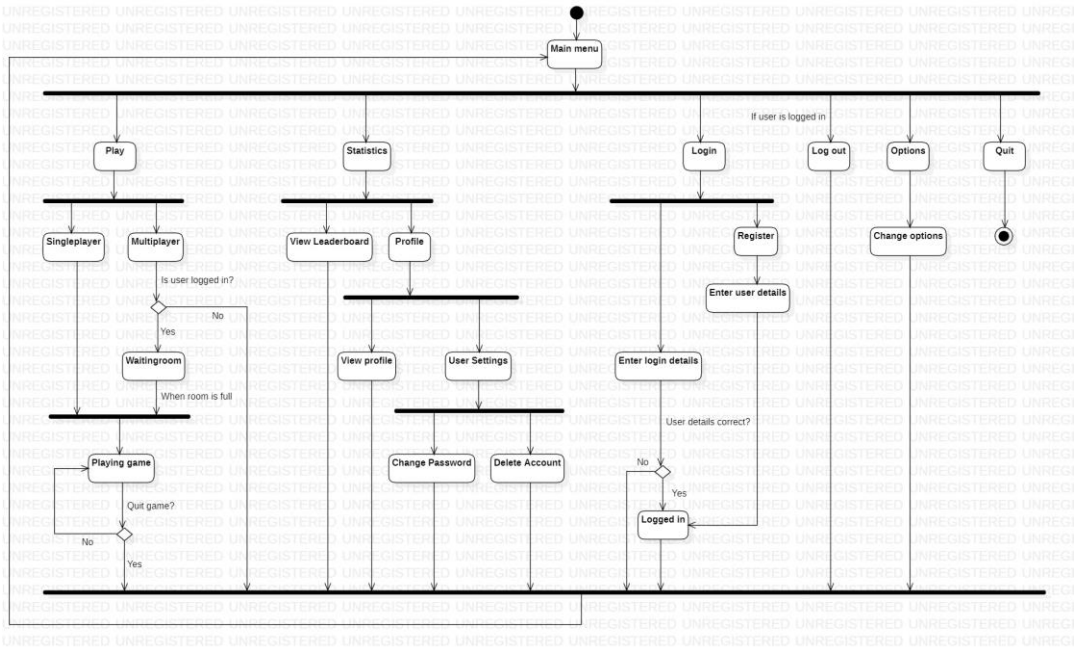
Koska React-komponentit ovat funktioita, niin minkään tasoista periytymistä ei ole, vaan kaikki on tehty koostamalla. Luokat esittävät kaavioissa React-funktiokomponentteja. Ainoa poikkeus on Routes ReactRouter -paketin alla, joka tosiasiassa on reitittimen konfigurointiin tarkoitettu objekti samassa tiedostossa

App-komponentin kanssa.



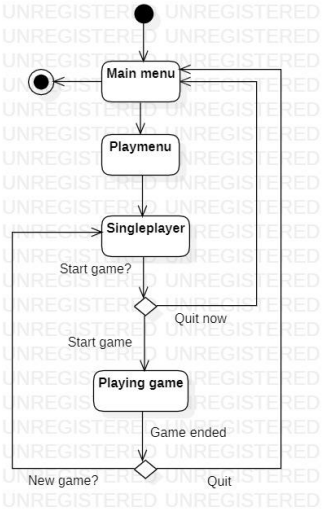
6 Ohjelmiston toiminta

Päävalikko



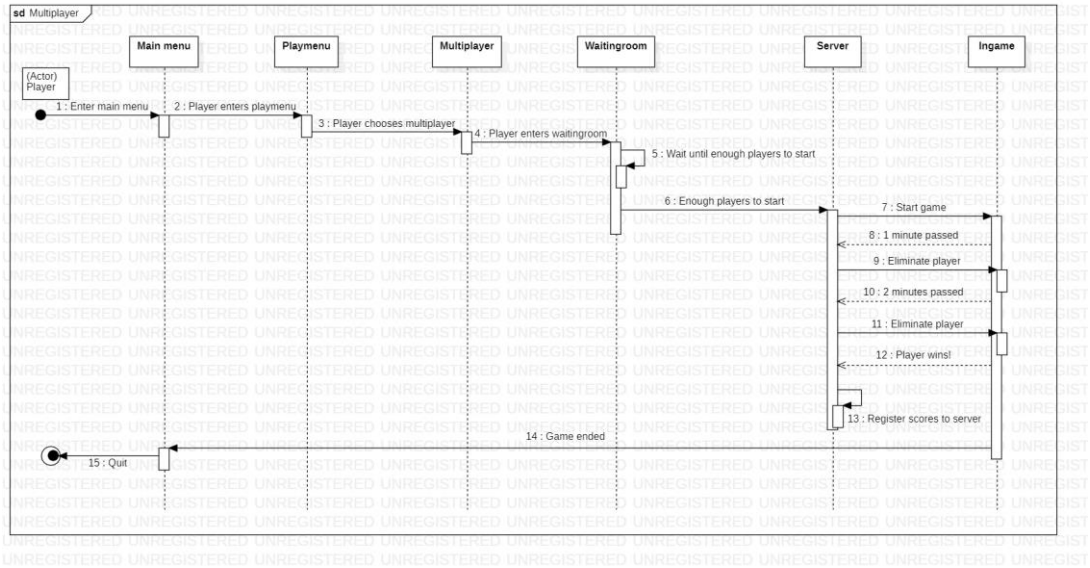
Aktiviteettikaavio päävalikon toiminnasta

Yksinpeli



Aktiviteettikaavio yksinpelitapahtumasta

Moninpeli



Esimerkki moninpelitapahtumasta kolmella pelaajalla sekvenssikaavion muodossa.

7 Kehitysprosessi ja kehitysvaiheen tekniikat

Pelin kehitysprosessi aloitettiin kirjoittamalla verkossa Google Docs -tekstidokumenttiin projekti-ideoita, joista yksi valittiin. Kun pelin idea oli päätetty, tehtiin kaavioita ohjelmiston tietomallille ja rakenteelle, joiden pohjalta luotiin ohjelmiston käyttöliittymää ja palvelinpuolta. Kaavioiden luomiseen käytettiin StarUML-työpöytäsovellusta ja Diagrams.net web-sovellusta.

Yhteydenpito ja palaverit pidettiin Discord-sovelluksen kautta muun muassa äänipuheluiden, ruudun jaon ja viestien avuin. Tärkeisiin projektiin liittyviin materiaaleihin vieviä linkkejä tallennettiin Discord-palvelimelle myöhempää käyttöä varten, helpottaen informaation saatavuutta. Projektiin kuuluvia työtehtäviä, työtunteja ja projektin yleistä kehitystä ylläpidettiin ja seurattiin Trello ja Nektion web-sovelluksissa.

Pelin käyttöliittymän kehittämiseen käytettiin Unity-kehitysalustaa ja ohjelmiston versionhallintaan Git:iä. Versioita pelin käyttöliittymästä tallennettiin GitHub-palveluun ja ohjelmiston palvelinpuolesta GitHub ja GitLab-palveluihin. Ohjelmiston palvelinpuolen kehitysympäristö jaettiin tiimiläisten kesken Docker-konteissa

GitHub:in kautta. Ohjelmiston koodi kirjoitettiin Visual Studio Code -sovelluksella C#-ohjelmointikielellä.

Palvelimella käytettiin MariaDB:tä SQL-tietokantana. Palvelimena .NET CORE 6 ASP.NET:iä, jossa käytettiin Entity Framework:ia tietokannan käsittelyyn. Palvelin toteutettiin RESTful-tyyppisenä. Tietokanta ja palvelin ovat molemmat omien Docker-konttien sisällä. Palvelin pyörii mukautetun Ubuntu-kuvan päällä ja MariaDB sen oman virallisen kuvan päällä.

Web Frontend oli Typescript + React projekti, jonka kehittämiseen ja rakentamiseen käytettiin Viteä. Yksikkötestaukseen Web Frontend:issä käytettiin Jest-kirjastoa. Rakennettu versio laitettiin tuotantoon Nginxillä hostaten, joka ajettiin sen omassa Docker-kontissa.

8 Lokalisaatio

Lokalisoinnin kieleksi saatiin japanin kieli. Kehityskielenä käytettiin englannin kieltä. Ennen prosessin alkua tutkittiin vastaavanlaisten sovellusten ja nettisivujen tyypillistä kielenkäyttöä. Tyypillisistä lauseista ja sanamuodoista tehtiin kokoelma, josta otettiin käyttöön tarvittavia sanoja tai lauseita. Lauseita mukautettiin projektiin sopiviksi, konsultoimalla tarvittaessa kielenpuhujia henkilökohtaisesti tai käyttämällä Reverso Context -nettisivua. Käännösprosessissa täytyi ottaa konteksti vahvasti huomioon ja tarvittaessa muuttaa sanoja, jotka eivät käännöskielellä toimineet.

Käännösprosessin alettua tutustuttiin ja opeteltiin Unity:n lokalisaation toimintaa. Projektiin luotiin asetukset “Localization Settings”-valikosta, josta saatiin luotua lokalisaation kielet. Tämän jälkeen luotiin merkkijonotaulu nimeltä “Game Table”, johon sijoitettiin pelin sisältämät merkkijonot. Pelissä oleviin tekstikomponentteihin asetettiin vastaavat merkkijonot taulusta. Merkkijonot käännettiin käyttämällä apuna aikaisemmasta tutkimuksesta tehtyä kokoelmaa ja hakemalla tarvittaessa uudestaan puuttuviin sanoihin käännöksiä. Merkkijonojen toimintaa testattiin

manuaalisesti itse pelieditorin kautta. Projektiversion vaihtuessa merkkijonojen asettelu tekstikomponentteihin tehtiin uudestaan ja tarkistettiin.

Nettisivu on lokalisoitu suomeksi, englanniksi ja japaniksi käyttämällä REACT Intl -lokalisaatio kirjastoa.

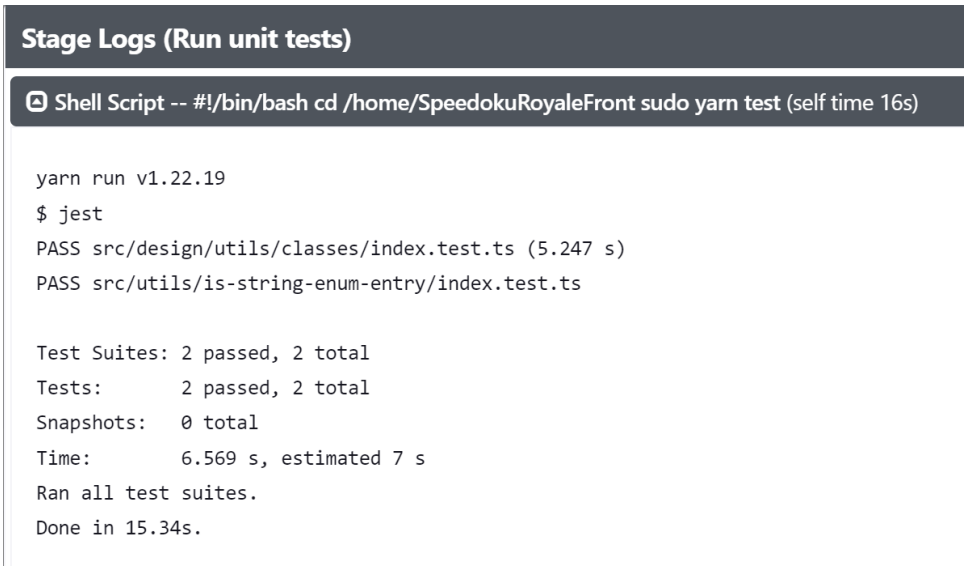
9 Testaus

Manuaalinen testaus

Manuaalista testausta tehtiin peliä pelaamalla aina muutosten jälkeen, ennen muutosten viemistä Unity-projektin Git-säiliöön. Lokalisaatio on testattu manuaalisesti, kuten edellisessä kappaleessa mainitaan.

Automaattitestaus

Web Frontend:issä yksikkötestejä ajettiin Jest-kirjastolla. Testit ajettiin Jenkins-livetysputken osana.



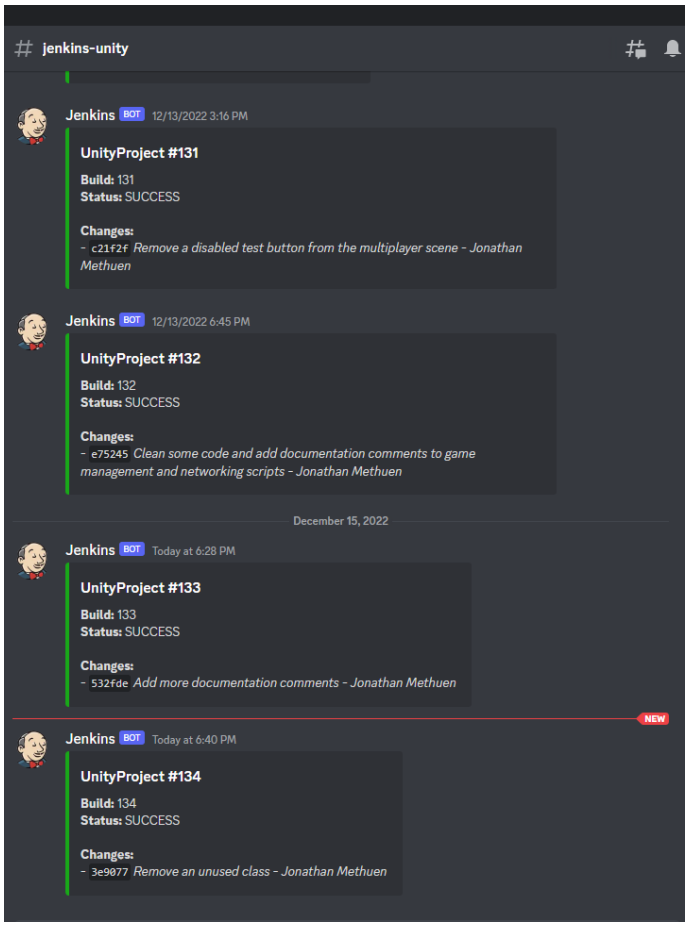
Näkymä Jenkins-putken logista kohdasta jossa testit ajetaan.

Pelin toimintaa testataan ”Unity Test Framework” -paketin avulla. Testeissä käydään läpi, löytyykö pelin kaikki tärkeät nappulat ja syöttökentät ja testataan, reagoiko peli oikealla tavalla nappien painalluksiin ja käyttäjän syöttämään tekstiin.

Testeistä käy ilmi seuraavat asiat:

- voiko pelaaja aloittaa yksinpelin
- löytyykö nappi moninpelin aloitukseen
- voiko yksinpeliä pelata
- voiko yksinpelistä poistua pelisession jälkeen tai kesken pelisession
- voiko käyttäjä luoda käyttäjätunnuksen
- voiko käyttäjä kirjautua sisälle luomallaan käyttäjätunnuksella
- voiko käyttäjä kirjautua ulos
- voiko käyttäjä vaihtaa käyttäjätunnuksensa salasanan ja kirjautua sisälle tällä uudella salasanalla
- voiko käyttäjä poistaa käyttäjätunnuksensa

Unity-projektin Jenkins-putki tarkistaa kahden minuutin välein onko pelin Git-repositorion oksissa tapahtunut muutoksia. Jos muutoksia on tapahtunut, Jenkins käynnistää Unity-projektin putken. Putkessa ajetaan pelin testit, julkaistaan testien tulokset Jenkins-sivulle ja rakennetaan pelistä koontiversio. Lopuksi Jenkins raportoi putken ajon tuloksen Discord-serverille omalle tekstikanavalleen, josta tuloksia on kätevä seurata.

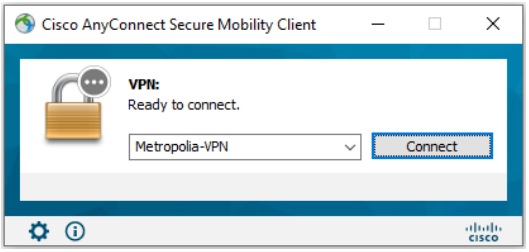


Näkymä Discord-kanavasta

10 Käyttöönotto

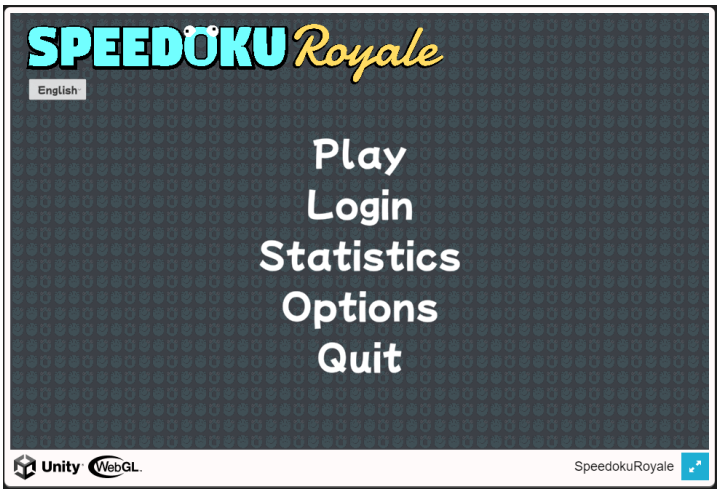
Pelin avaaminen nettiselaimessa:

1. Varmista, että olet kirjautunut Metropolia-VPN palveluun.



2. Syötä selaimesi osoitekenttään seuraava osoite: http://10.114.32.14/play

3. Saavut sivulle, jossa on WebGL-kehys, jonka pitäisi avautua automaattisesti sivun avattua. Voit aloittaa pelin pelaamisen, kun kehys on suorittanut latauksen.



11 Yhteenveto ja jatkokehitys

Projektin tarkoitus oli tarjota pelaajille ohjelmisto, jolla voi pelata aikarajoitettuja sudokuja niin yksin, kuin muitakin pelaajia vastaan. Dokumentissa mainitsemamme tavoitteet toteutuivat hyvin suhteessa siihen, kuinka paljon aikaa projektin työstämiseen oli. Unityllä toteutettu käyttöliittymä mahdollistaa pelin levittämisen tarvittaessa monelle eri alustalle ja Docker-konteissa toimiva palvelinpuoli on helposti siirrettävissä ja käyttöönotettavissa eri hosteissa. Alunperin ohjelma toteutettiin Windows-alustalle, mutta lopulta kehityksessä siirryttiin WebGL-alustalle, jotta pelin voisi käynnistää nettisivun välityksellä. Projektin kehitys tapahtui käyttäen Git-versionhallintaa GitHubissa ja GitLabissa hostattuna. Nektionissa ja Trellossa noudatettiin Scrum-työtapaa sprinttien muodossa.

Projektin toteutuksessa jäi puutteellisiksi käyttäjätietojen ja peliasetuksien muuttamisen toiminnot, kuten käyttötapausten listauksesta voidaan todeta. Tulevaisuudessa sovellukseen voitaisiin kehittää loppuun myös palkintojärjestelmä, jonka avulla pelaajia palkittaisiin voitoista profiilin personointi toiminnallisuuksilla. Peliin on suunniteltu äänimaailmaa, joka olisi mahdollista ottaa käyttöön. Pelin ulkonäköä voitaisiin elävöittää animoiduilla grafiikoilla.