Aki Koppinen – Ville Riepponen – Jere Lampola – Tuomas Rajala – Tatu Nordström

OTP-2 Projektin dokumentaatio

Metropolia Ammattikorkeakoulu
Tieto- ja viestintätekniikka
Ohjelmistotuotanto
Ohjelmistotuotantoprojekti 2
06.05.2021

Sisällys

1 Johdanto

Projektiryhmä, johon kuuluu Aki Koppinen, Ville Riepponen, Jere Lampola, Tuomas Rajala ja Tatu Nordström, sai Ohjelmistotuotantoprojekti 1 (OTP1) ja Ohjelmistotuotantoprojekti 2 (OTP2) kurssien aikana valmiiksi AutoMatti nimisen sovelluksen. Tämä sovellus on tarkoitettu peliympäristöksi erilaisille automaatti –tyylisille peleille joista pokeri on lopullisessa versiossa täysin pelattavissa. Molemmilla opintojaksoilla harjoiteltiin ketterän projektin läpiviemistä Scrum-työtapaa noudattaen. Ryhmä käytti projektin kulun suunnitteluun ja seurantaan Nektion -verkkosovellusta, johon kukin ryhmän jäsen merkkasi tehtyjä tehtäviään ja kirjasi työtuntejaan.

OTP2 kurssilla jatkettiin ohjelmiston kehittämistä siitä mihin OTP1 kurssilla jäätiin. Uusia näkökulmia OTP2 opintojaksossa olivat muun muassa tuotteen lokalisointi ja kansainvälistäminen.

Tämä dokumentti toimii AutoMatti -ohjelmistomme loppuraporttina ja dokumentaationa jatkokehitystä ja sovelluksen käyttämistä varten. Dokumentissa käydään läpi sovelluksen ideaa, teknistä toteutusta, toteutuneita toiminnallisuuksia sekä jatko kehityskohteita.

2 Visio

Tavoite: Selkeä ja helppokäyttöinen peliympäristö, joka tarjoaa riskittömän tavan pelata erilaisia pelejä, kuten automaattipokeria. Peli pitää yllä listaa varakkaimmista pelaajista, joten asiakkaat voivat tavoitella parasta sijaa.

Asiakkaat: Pokerista ja muista automaattipeleistä kiinnostuneet kilpailuhenkiset henkilöt.

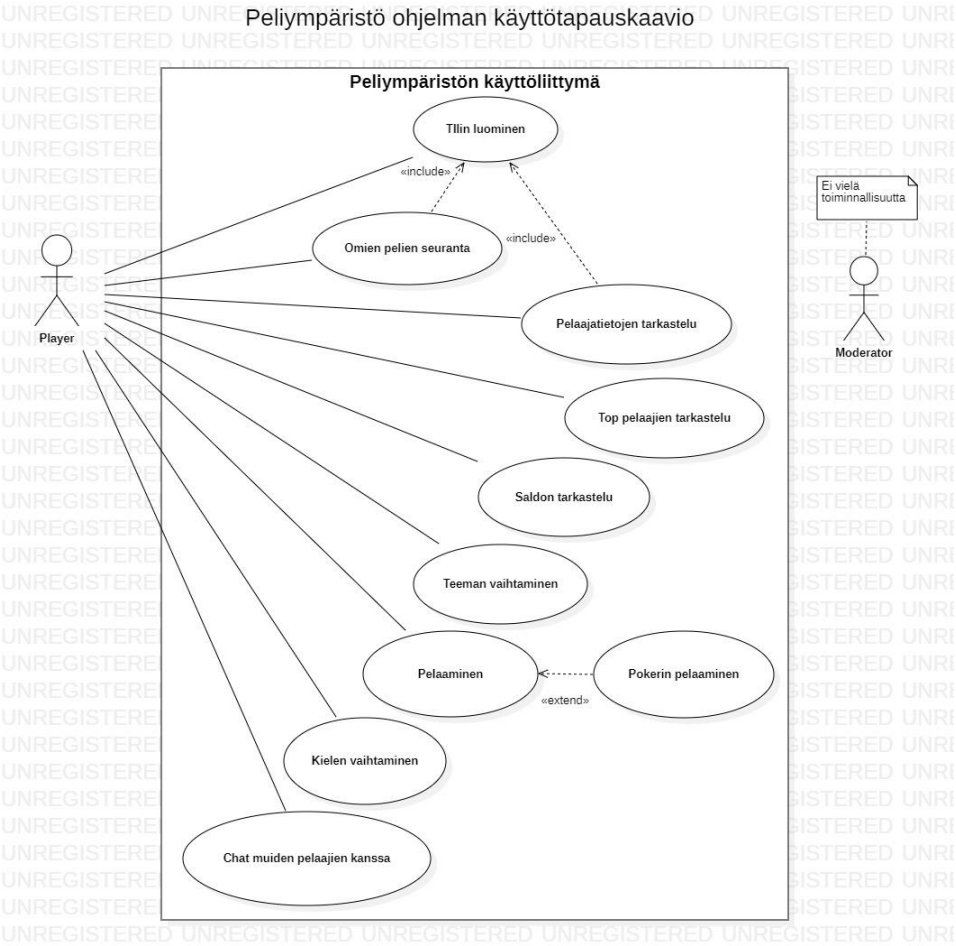
Tarve: Korona aikana ihmiset tarvitsevat vaihtoehtoja suljetuille huoltoasemien ja lähikauppojen automaattipeleille.

Kilpailu: Sovellus on ilmainen ja kaikkien käytettävissä. Sitä voi pelata omalta kotikoneeltaan poistumatta kotoa. Oikean rahan menetyksen riskiä ei ole.

Erilaistaminen: Käyttäjä voi muokata peliympäristöä erilaisilla teemoilla haluamansa laiseksi. Chat -ominaisuus mahdollistaa pelaajien välisen sosialisoimisen.

3 Toteutettu toiminnallisuus

Peliympäristössä saatiin toteutettua toimivat tietokantayhteydet, käyttöliittymä ja pokeripeli. Käyttäjän toiminnot on kuvattu alla olevassa käyttötapauskaaviossa (Kuvio 1). Siitä on nähtävissä ne toiminnot mitkä ovat tällä hetkellä käyttäjän, eli pelaajan, käytettävissä. Osa näistä vaatii tilin luontia ja kirjautumista, mikä on kuvattu yhteydellä Tilin luomiseen. Pokerin lisäksi on suunniteltu myös muita pelejä Peliympäristöön, joten pokerin pelaaminen on vain osa pelaamista kokonaisuudessaan. Moderaattori on kuvattu toisena käyttäjänä, vaikka toiminnallisuutta ei varsinaisesti ole vielä sille luotu.



Kuvio 1. Käyttötapauskaavio

3.1 Tietokanta

Tietokantaan on mahdollista tallentaa pelaajatietoja ja pelattuja pelejä. Pelaajatiedot tallentuvat, kun käyttäjä rekisteröi tunnukset ja hän voi myös muokata pelaajatietojaan.

Pelatut pelit tallentuvat aina tietokantaan kun peli saadaan päätökseen. Näitä pystytään myös hakemaan tietokannasta ja niistä voidaan muodostaa tilastoja, joita sitten voidaan näyttää käyttäjälle.

3.2 Käyttöliittymä

Käyttöliittymässä on mahdollisuus siirtyä pelaamaan pokeripeliä, vaihtaa asetuksia ja seurata pelitilastoja. Asetuksissa on mahdollista vaihtaa sovelluksen kieltä Suomen ja Englannin välillä, sekä vaihtaa käyttöliittymän teemaa. Tilastoissa voi päästä näkemään listauksen eniten saldoa keränneistä pelaajista ja kirjautunut käyttäjä voi myös nähdä omat pelitilastonsa ja rahan kehityksen.

Käyttöliittymässä tarjotaan yläpalkissa myös mahdollisuus luoda tili, kirjautua sisään sekä kirjautua ulos. Yläpalkin näkymä muuttuu riippuen siitä, onko käyttäjä kirjautunut sisään vai ei. Käyttäjä saa virheilmoituksia vääristä syötteistä kirjautumisessa ja rekisteröinnissä, sekä hänelle kerrotaan miten hän voi korjata virheensä. Käyttäjä voi myös siirtyä tarkastelemaan omia tietojaan ja muuttaa samalla pelaajatunnusta ja salasanaa.

Käyttöliittymän chat -ikkuna mahdollistaa pelaajien välisen kommunikaation. Chat ikkuna aukeaa sovelluksen vasemmasta alareunasta ja on saatavilla jokaisessa sovelluksen näkymässä.

3.3 Pokeripeli

Kirjautumista ei vaadita pokeripelin pelaamista varten. Käyttäjä voi siirtyä suoraan pelaamaan ilman kirjautumista ja jos hän myöhemmin niin haluaa, hän voi siirtää kerätyt rahat uudelle tilille rekisteröinnin yhteydessä.

Pokeripeli itsessään on saatu pelilogiikaltaan varsin toimivaksi ja se pystyy jakamaan pelaajalle oikean määrän kortteja ja ilmoittamaan voittavat kädet. Käyttäjä voi panostaa peliin haluamansa verran rahaa, joka vaikuttaa saldon määrän muutokseen pelin päättyessä. Ensimmäisen jaon jälkeen pelaaja voi lukita haluamansa kortit lisätäkseen voittomahdollisuuksia. Lukitsemisen yhteydessä kortit muuttuvat harmaiksi ja saavat lukon kuvan. Häviävä käsi muuttuu kauttaaltaan harmaaksi ja voittava käsi saa vihreän ympäryksen. Pokeripeli on grafiikaltaan selkeä ja siinä käytetään siistittyjä pelikorttikuvia.

Voittavan käden saadessaan pelaajalla on mahdollisuus yrittää lisätä voittojaan tuplausvalinnalla. Tuplatessa pelaaja yrittää arvata joko seuraavan kortin arvoa (pieni A-6, suuri 8-K) tai maata. Oikea arvo tuplaa voiton ja oikea maa nelinkertaistaa voiton. Pelaaja voi jatkaa uhkapeliä uusilla arvauksilla, tai siirtyä takaisin pelinäkymään ja valita voitonmaksun, jolloin voitot tallentuvat käyttäjälle. Pelaajan arvatessa väärin, voitot menetetään ja pelaajalta vähennetään panoksen verran saldoa.

3.4 Lokalisaatio

Sovellusta on mahdollista käyttää englannin- sekä suomen kielellä. Asetukset sisältää kielivaihtoehdot ja kielen valitessa käyttöliittymäkomponenttien tekstit käännetään valitulle kielelle sekä numeeriset arvot formatoidaan kohdekielen mukaan. Sovelluksen pääkohderyhmän ollessa suomalaiset, päätimme pitää oletuskielenä suomen. Englannin lisäys mahdollistaa sovelluksen kansainvälistämisen sekä lisää sovelluksen saavutettavuutta (Kuvio 2).

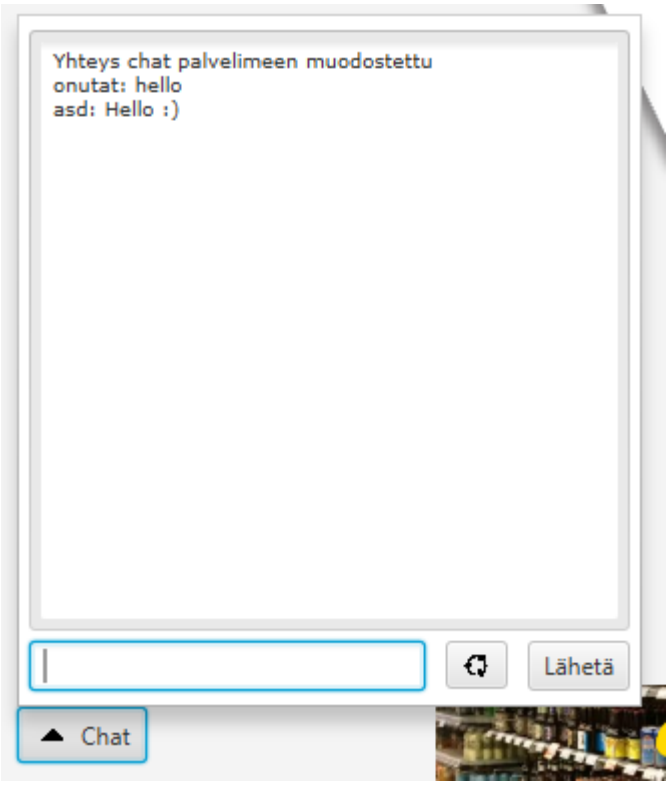
#	Saldo	Saldon mu...	Voitto/Häviö
215	802,00	-1,00	Häviö
214	803,00	-1,00	Häviö
213	804,00	-1,00	Häviö
212	805,00	-1,00	Häviö
211	806,00	-1,00	Häviö
210	807,00	-10,00	Häviö
209	817,00	70,00	Voitto
208	747,00	50,00	Voitto
207	697,00	-10,00	Häviö
206	707,00	-10,00	Häviö
205	717,00	-10,00	Häviö
204	727,00	30,00	Voitto
203	697,00	-10,00	Häviö
202	707,00	-10,00	Häviö
201	717,00	-10,00	Häviö
200	727,00	-10,00	Häviö
199	737,00	-10,00	Häviö

#	Balance	Balance ch...	Win/Loss
215	802.00	-1.00	Loss
214	803.00	-1.00	Loss
213	804.00	-1.00	Loss
212	805.00	-1.00	Loss
211	806.00	-1.00	Loss
210	807.00	-10.00	Loss
209	817.00	70.00	Win
208	747.00	50.00	Win
207	697.00	-10.00	Loss
206	707.00	-10.00	Loss
205	717.00	-10.00	Loss
204	727.00	30.00	Win
203	697.00	-10.00	Loss
202	707.00	-10.00	Loss
201	717.00	-10.00	Loss
200	727.00	-10.00	Loss
199	737.00	-10.00	Loss

Kuvio 2. Esimerkki lokalisaatiosta pelihistoriasta.

3.5 Chat

Sovellus sisältää chat –ominaisuuden (Kuvio 3), joka mahdollistaa pelaajien välisen kommunikoinnin reaaliajassa. Chatin lisäämisellä pyritään imitoimaan reaalimaailman automaattipelikokemusta, koska useille pelaajille pelaaminen on myös sosiaalinen tapahtuma. Chatin toiminta vaatii tällä hetkellä sen, että sovelluksen kehitysryhmän jäsen käynnistää chat-palvelimen.



Kuvio 3. Chat ikkuna

4 Jatkokehitysideat

Ohjelmisto on pokeripelaamisen osalta saatu varsin vakaaseen tilaan. Chat ominaisuuden toteutuksen myötä tiimimme sai käsityksen verkkoyhteyksien luomisesta käyttäjien välille Javalla, joka mahdollistaa yksinkertaisten moninpelin toteuttamisen. Seuraaviin kohtiin on kasattu jatkokehitysideat muokkauksille ja lisäyksille ohjelmistoon.

4.1 Muokkaukset

Lokalisaation kattavuuden parantaminen ohjelmistossa.

Käyttäjänimien väritys eri väreillä chatissa.

Oman profiilin korostus tilastonäkymässä.

Online pelaajamäärän toiminnallisuuden loppuun vieminen.

Mainosten vaihtuvuus.

4.2 Lisäykset

Pokeripelissä tulisi olla jokerikortteja.

Uusien kielien lisääminen lokalisointiin.

Käyttäjälle tulisi tehdä mahdolliseksi pelata pokeripeliä moninpelinä.

Moninpelaamiselle tulee olla odotushuone, jossa voi odottaa pelin alkamista.

Peliympäristöön tulee lisätä muita pelejä.

Sovellukseen lisätään moderaattori, joka pystyy seuraamaan tilastoja ja vaikuttamaan huonosti käyttäytyvien pelaajien toimintaan.

Käyttäjien tulisi voida antaa palautetta kehittäjille.

Käyttöliittymää tulee voida muokata myös muilta osin kuin vain teeman suhteen (ääni-maailma).

Käyttäjätilin palautus salasanan unohtuessa.

5 Ohjelmointitekninen toteutus

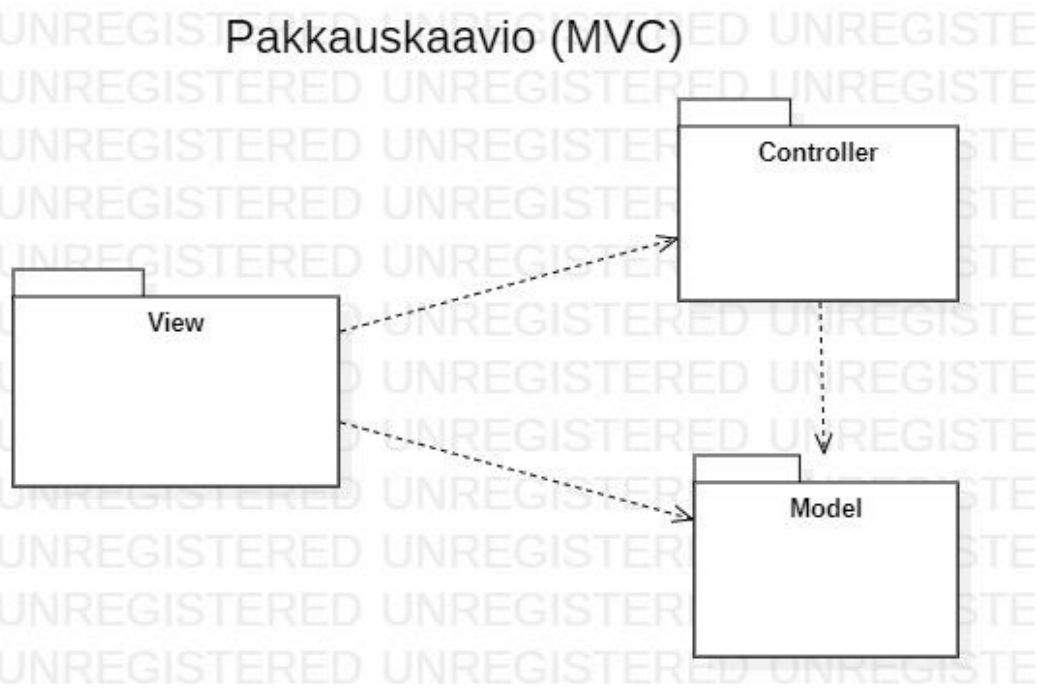
5.1 Käytetyt ohjelmointikielet ja kirjastot (ulkoiset API:t).

Ohjelmisto on toteutettu kokonaisuudessaan Java-ohjelmointikielellä. Käytetty versio JDK:sta on 1.8.0_261.

5.2 Arkkitehtuuri

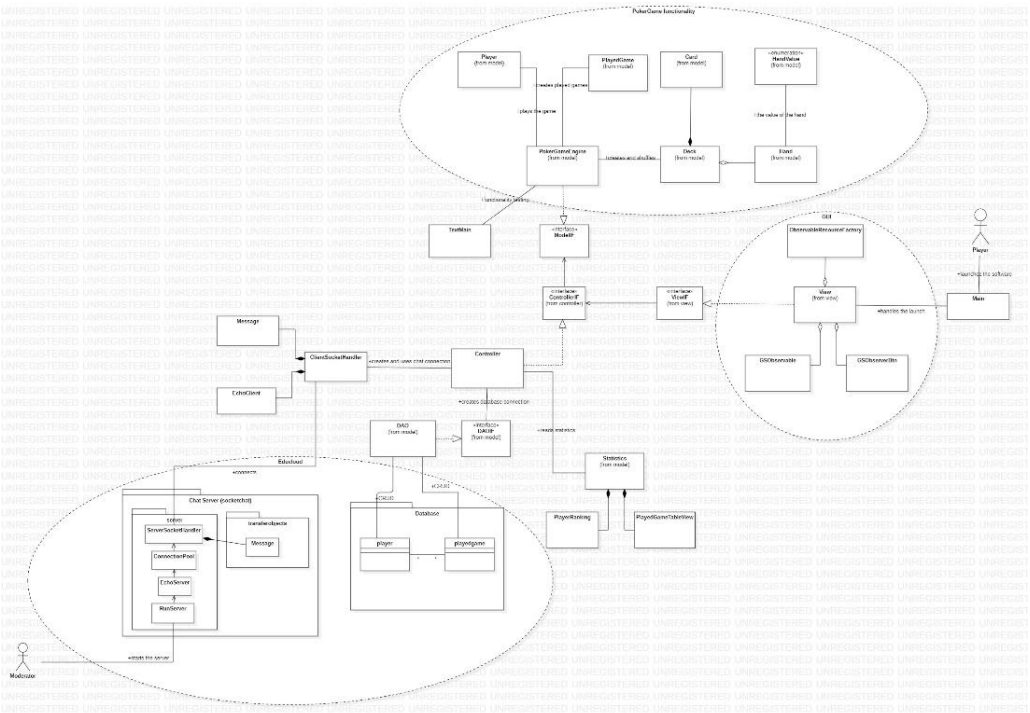
Ohjelmisto toteuttaa MVC mallia, jossa graafinen liittymä on View ja pelattava peli on Model. Controller yhdistää nämä kaksi toiminnallisuutta. Nettiyhteyttä vaativat toiminnallisuudet eli tietokanta ja chat palvelin ovat myös sidottuina suoraan Controller luokkaan. Näin on varmistetaan että yhteyksien luonti tietokantaan ja palvelimeen onnistuu riippumatta onko valittu peli käynnissä vai ei.

MVC mallin mukaisesti tietoa vaihdetaan View ja Model pakkauksien välillä. View luokkaan on kuitenkin tallennettu Model pakkauksen Player olio. Tämä helpotti kirjautuneen käyttäjän tunnistamista, johon kyseistä luokkamuuttujaa käytetään. Tämä tekee MVC rakenteesta pehmeän jossa View pakkauksesta on myös suora yhteys Modeliin (Kuvio 4).



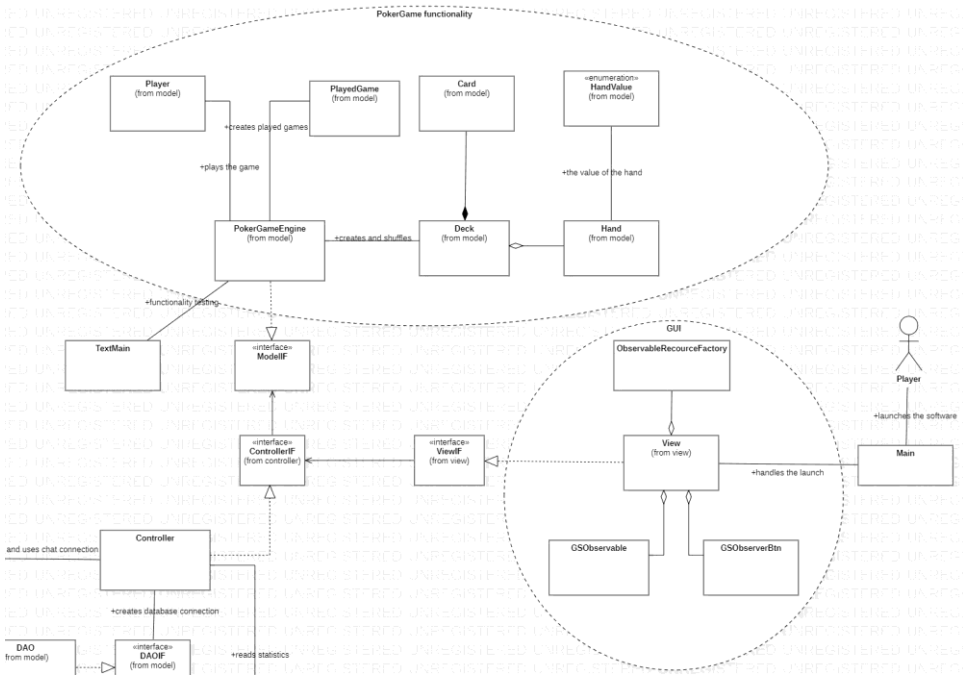
Kuvio 4. Ohjelmiston pakkausdiagrammi, pehmeämpi versio MVC-mallista.

Pelaaminen, käyttöliittymä, tietokantayhteydet ja chat-palvelinyhteydet on kaikki ohjelmistoarkkitehtuurissa erotettu omiksi kokonaisuuksikseen. Nämä kaikki ovat MVC mallin mukaisesti yhteydessä toisiinsa Controllerin ja sen rajapinnan ControllerIF kautta (Kuvio 5).



Kuvio 5. Ohjelmiston arkkitehtuuri

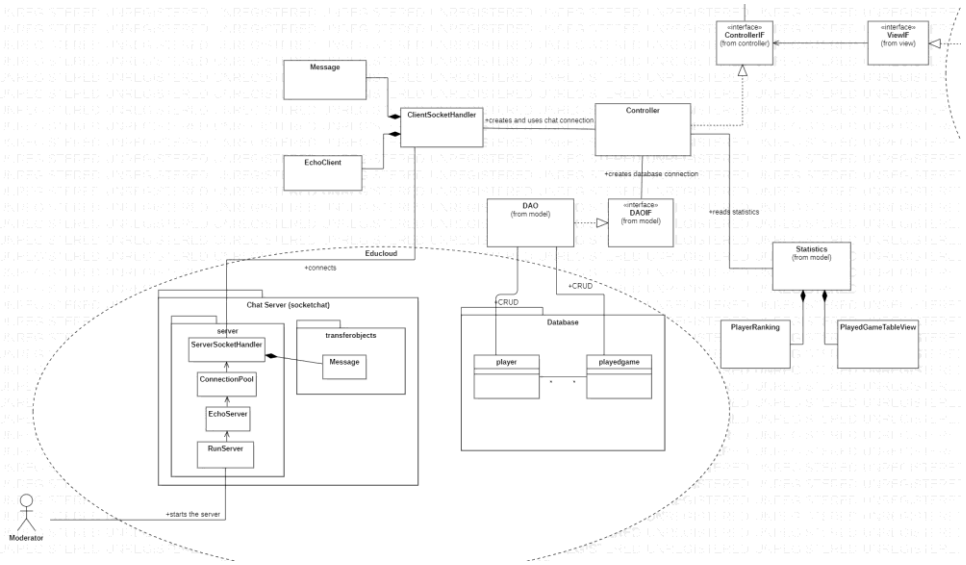
PokerGameEngine toteuttaa Model toiminnallisuutta MVC mallissa ModelIF rajapinnan kautta, ja se vastaa kaikesta pokeripelin toiminnallisuudesta. Uuden pelin lisääminen ohjelmistoon vaatisi uuden GameEngine luokan joka toteuttaisi myös ModelIF rajapintaa. GameEngine luokkia voisi sitten vaihtaa ajonaikaisesti käyttäjän valitessa toisen pelin. PokerGameEngine luokkaan yhdistetyt PlayedGame ja Player luokat ovat Hibernaten kanssa tietokantaan yhdistettyjä luokkia jotka vastaavat pelatun pelin tietojen ja pelaajatietojen viennistä tietokantaan. Tämän vuoksi myös uuden GameEngine luokan tulee liittää ne osaksi toteutustaan (Kuvio 6).



Kuvio 6. Ohjelmiston arkkitehtuuri, Pelilogiikka ja käyttöliittymä

Käyttöliittymä on luotu suureksi osaksi View-luokkaan jolla on apuluokkina GSObservable, GSObserverBtn ja ObservableRecourceFactory. GSObservable -luokassa säilytetään pokeripelin tilaa, ja luokkaan on kuuntelijoiksi liitetty GSObserverBtn luokan toteuttavia pelin painikkeita. Pelin painikkeet toimivat eri tavalla pelin edetessä ja Observable -malli helpottaa toimintojen toteutusta. Pelin kieliasetukset on listattu avain – arvo pareina resurssitiedostoihin, joita varten on luotu ObservableRecourceFactory. Luokka mahdollistaa StringBindingien luonnin sovelluksen tekstikomponenteille, jolloin komponentit kuuntelevat resurssien muutosta ja osaavat vaihtaa tekstinsä automaattisesti kielen vaihtuessa ilman näkymien uudelleenlataamista.

Tietokanta ja chat-palvelin on yhdistetty suoraan Controlleriin, koska tilastoja ja keskustelu toiminnallisuutta piti pystyä käyttämään myös silloin, kun mikään peli ei ollut käynnissä. Nyt tietokantayhteyksistä vastaava DAO luokka voidaan viedä eri pelien käytettäväksi Controllerista ja ClientSocketHandler luokka pystyy aina yhdistämään chat palvelimeen. Tietokannan ja chat palvelimen toiminnallisuus on ulkoistettu Educloud palvelimelle (Kuvio 7).



Kuvio 7. Ohjelmiston arkkitehtuuri, Tietokanta ja Chat-palvelin

Koska tilastojen haku ei tarvinnut enää kulkea PokerGameEnginen kautta, on tilastoista vastaava Statistics luokka yhdistetty luokkakaaviossa suoraan Controlleriin. Statistics luokalla on myös olemassa kaksi apuluokkaa (PlayerRanking ja PlayedGameRanking) jotka ovat kuitenkin olemassa vain tietokantahakuja varten.

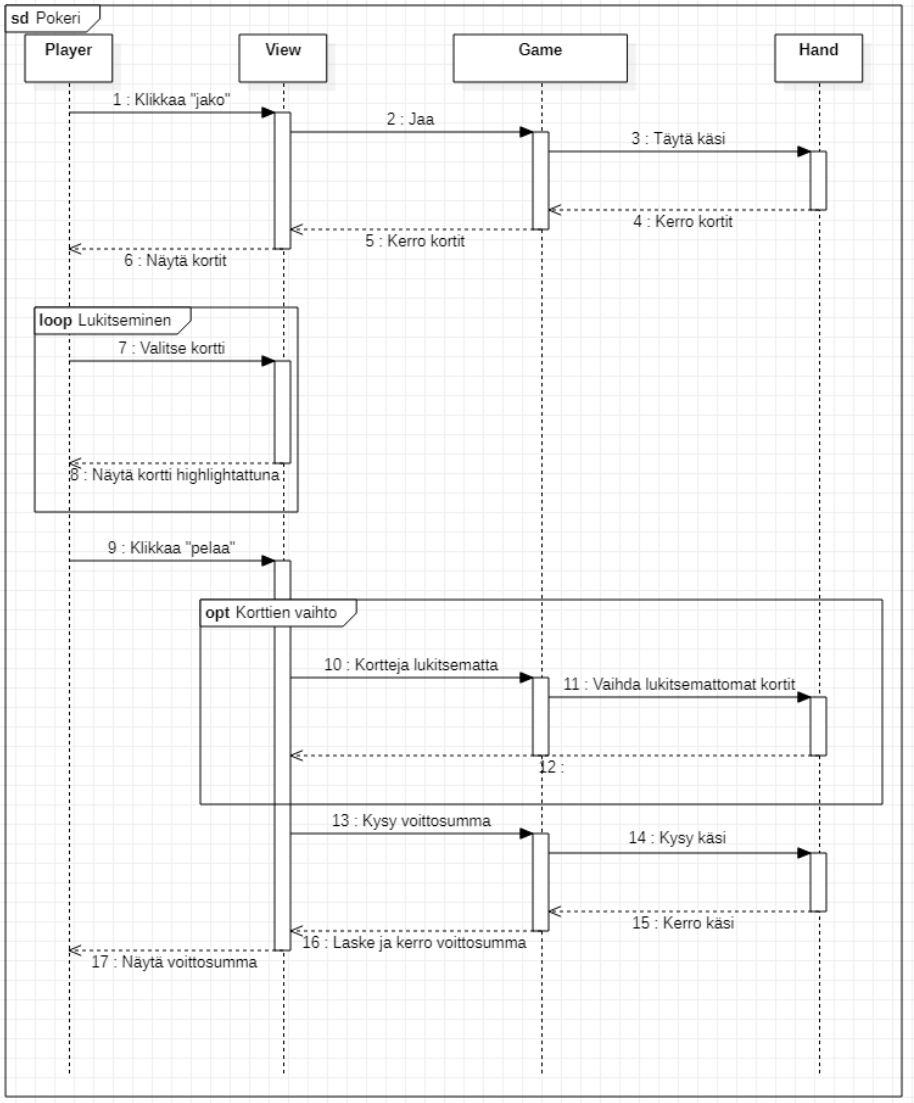
5.3 Käyttöliittymän kuvaus

Käyttöliittymän toteutus tapahtui JavaFX:llä joka sisältyy valittuun JDK versioon. Kaikki käyttöliittymä komponentit päätettiin laittaa yhteen ViewIF rajapintaa toteuttavaan View-luokkaan, josta kontrolleri joko haki tietoja tai jota kontrolleri päivitti. Eri elementtiryhmille on tehty omat Builder –metodit, jotka luovat omat elementtiryhmänsä JavaFX komponenttien avulla. Pokerissa käytetyt kuvat saatiin osoitteesta http://acbl.mybigcommerce.com/52-playing-cards/

5.4 Sisäisen logiikan kuvaus

Pokeripelissä näkymä ottaa vastaan pelaajan tekemät toiminnot ja välittää ne kontrollerin kautta pelimoottorille. Pelimoottori keskustelee pelaajalle jaettuja kortteja kuvaavan Hand-luokan kanssa, jonka tiedot välitetään takaisin kontrollerin ja näkymän kautta pelaajalle. Pokeripelilogiikka kuviossa (Kuvio 8) on esitetty pokeripelin loogista toimintaa tarkemmin ilman Controlleria joka olisi aina View ja Game (PokerGameEngine) luokkien välissä. Pelaajan antamat komennot siirtyvät pokeripelin aikana Viewistä (Controllerin

kauttta) PokerGameEngine luokalle, joka ottaa yhteyttä Hand luokkaan tehdäkseen muutoksia viiden kortin luetteloon. Tämä toimivuus toistuu pelaamisen eri vaiheissa.



Kuvio 8. Pokeripelin logiikka, Controller-luokka karsittu pois

View luokassa säilytetään oliomuuttujaa model pakkauksen Player luokasta, jota käytetään ilmaisemaan kullakin hetkellä kirjautunutta käyttäjää. Player olio käydään tietokannasta Controllerin, ja edelleen DAO luokan kautta. Muutokset pelaajan tilassa (saldon muutos/pelaaja tietojen muokkaus) aiheuttavat päivityksen tietokantaan ja View luokan Player olio-muuttujaan. Uloskirjautuminen ja sisäänkirjautuminen vaihtavat tilaa vain View luokan Player oliolle.

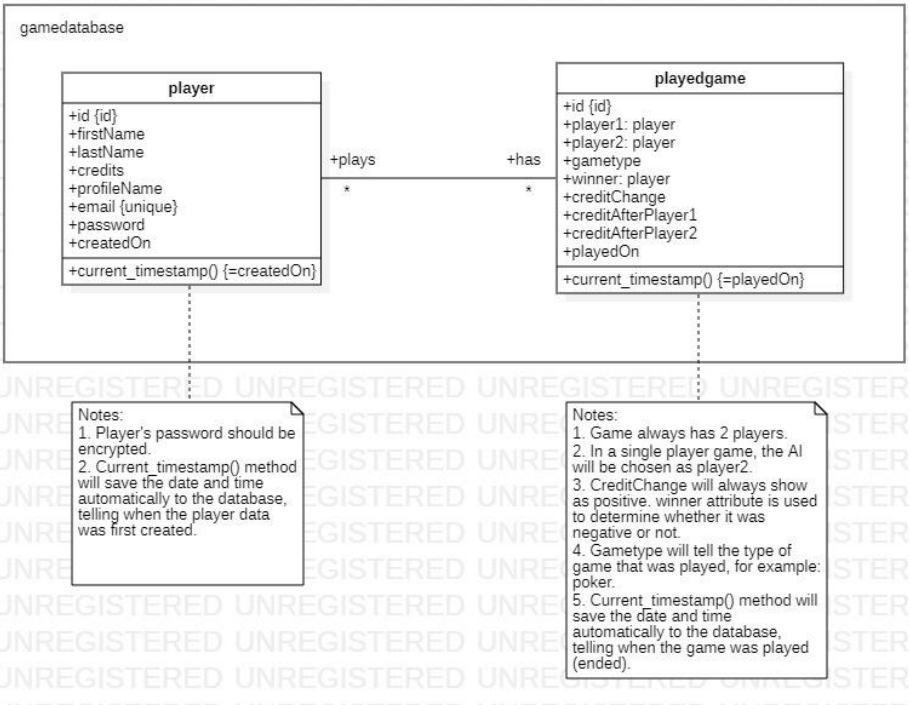
5.5 Ulkoisten tietovarastojen (tiedostot, tietokannat) kuvaukset

Tietokantayhteyksissä käytetään hyväksi Hibernate ORM -työkalua, mikä helpottaa datan hakua ja tallentamista.

Visuaalisessa käyttöliittymässä käytetään internetistä saamia open-source kuvatiedostoja pelikorteille sekä ikoneille.

Tietokanta on toteutettu MySQL:ään pohjautuvalla relaatiotietokantajärjestelmällä, MariaDB:llä. Se sijaitsee Metropolian omalla educloud virtuaalipalvelimella. Jotta tietokantaan saadaan yhteys, täytyy liikenne tunneloida virtuaalipalvelimelle. Tietokanta kulkee nimellä 'gamedatabase' ja se käsittää kaksi relaatiotaulukkoa; 'player' ja 'playedgame'. Nämä relaatiotaulut on esiteltyinä muuttujineen tarkemmin Tietokannan luokkakaaviossa (Kuvio 9). 'Player' -tauluun säilötään pelaajan etu- ja sukunimi, käyttäjätunnus, saldo, sähköpostiosoite, salasana sekä rekisteröitymisaika. 'Playedgame' -taulukkoon tallennetaan pelityyppi, kahden eri pelaajan id, voittajan id, saldon muutos, pelin päättymisaika sekä pelaajien saldo pelin jälkeen. Rekisteröitymisaika ja pelin päättymisaika tallentuvat automaattisesti molemmissa relaatiotauluissa olevilla current_timestamp() metodeilla.

Tietokanta luokkakaavio

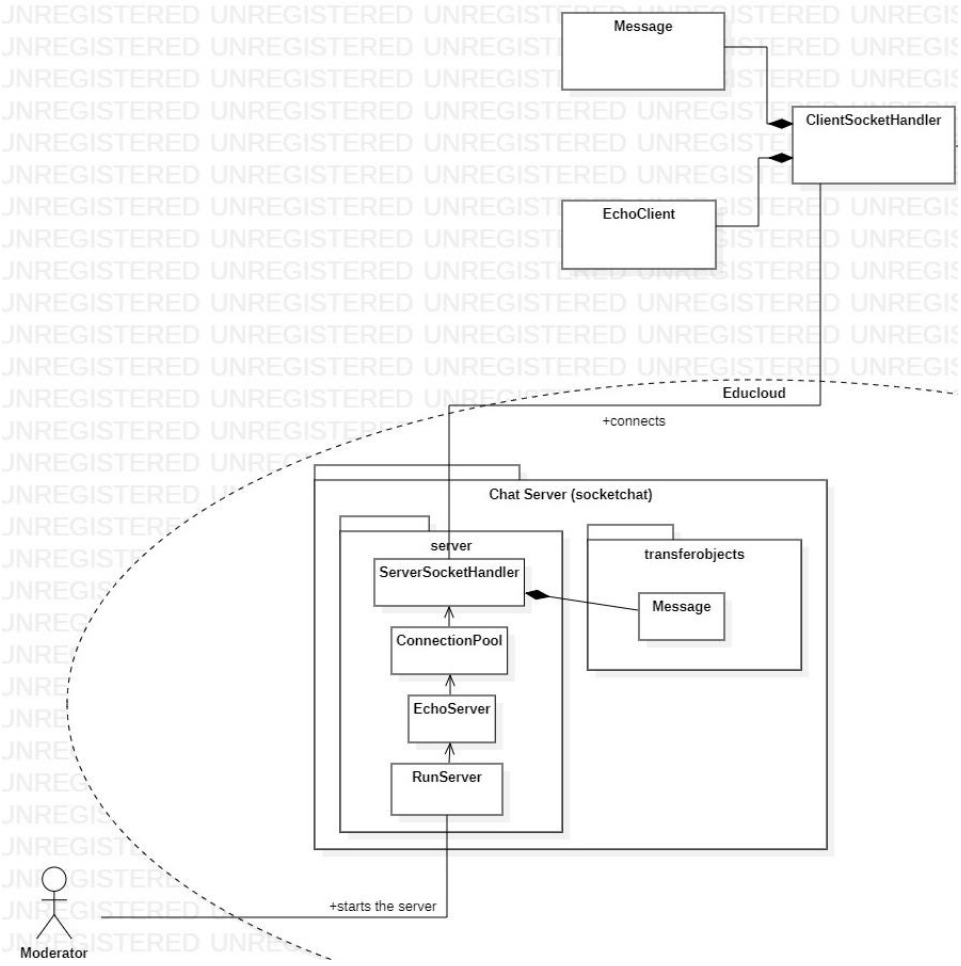


Kuvio 9. Tietokannan luokkakaavio

Pelatun pelin player1 vaihtuu sen mukaan, kuka on kirjautuneena sisään ohjelmistoon. Jos käyttäjä ei ole kirjautunut, käytetään Tester käyttäjää (tietokannassa id=1001). Näin kaikki pelatut pelit saadaan tallennettua tietokantaan. Tulevaisuuden kannalta päätimme jo mahdollistaa moninpelin hyvässä vaiheessa, joten myös player2 on osa playedgame relaatiotaulua, yksinpelissä player2 paikan ottaa AI (tietokannassa id=1000), joka on Testerin tavalla Peliympäristön kehittäjien itse toiminnallisuutta varten tietokantaan luotu Player olio.

5.6 Chat-palvelin

Uutena ominaisuutena sovellukseen on kehitetty keskustelukanava, jonka avulla pelaajat voivat pitää yhteyttä toisiinsa. Palvelin on sijoitettu Educloudiin ja sen käynnistää sovelluksen moderaattori (Kuvio 10).

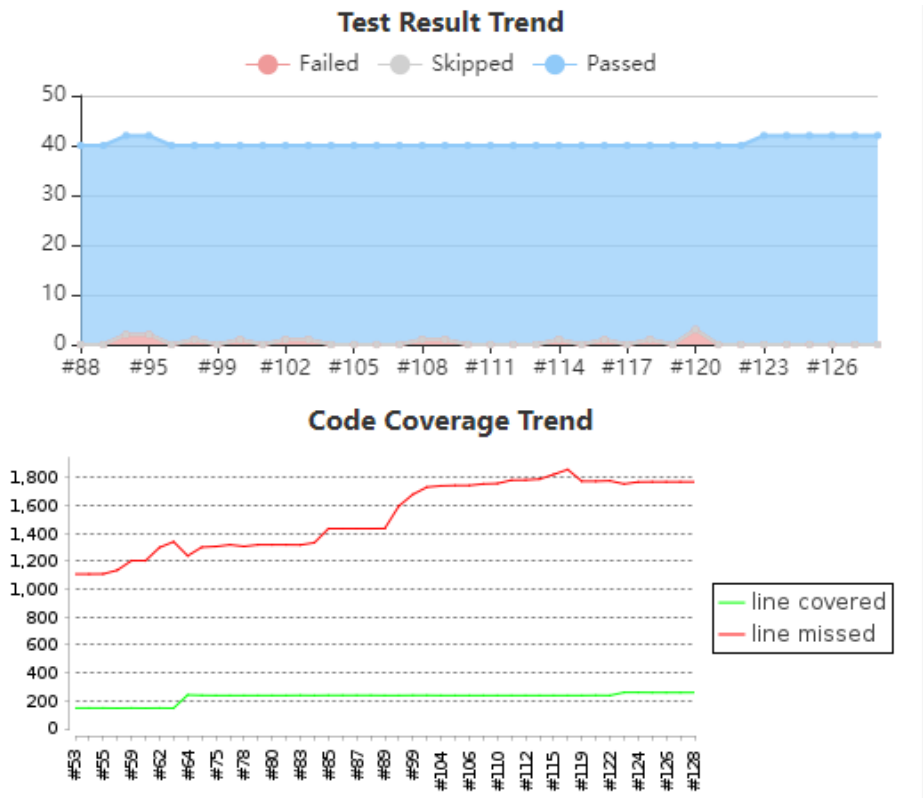


Kuvio 10. Chat-palvelimen arkkitehtuuri

Kun käyttäjä avaa sovelluksen, se luo uuden säikeen joka ottaa yhteyden palvelimeen. Palvelimessa ylläpidetään luotujen säikeiden listaa, ja uuden viestin ilmaantuessa se edelleenlähettää viestin jokaiselle säikeelle. Jotta lähetettyjen sekä vastaanotettujen viestin rakenne täsmäisi toisiaan, on niiden oltava samoja myös pakkauskohtaisesti. Mikäli palvelinyhteys katkeaa sovelluksen ajon aikana, on yhteyden palauttaminen mahdollista manuaalisesti käyttäjän toimesta.

5.7 Testaus

Teimme sovelluksen model -pakkauksen muutamille luokille yksikkötestauksia JUnit 5 kirjastoa hyväksikäyttäen. Jenkins -palvelin piti huolen, että testit ajettiin läpi aina, kun versionhallintaan tuli muutoksia. Testikattavuus ilmenee kahdesta graafista (Kuvio 11). Jenkins palvelin löytyy osoitteesta http://10.114.32.61:8080/ . Palvelimelle kirjautuminen vaatii tunnukset ja metropolia VPN yhteyden.



Kuvio 11. Testikattavuus graafina

Sovelluksen ja tietokannan välistä toimintaa päätimme testata manuaalisesti konsolistulosteiden avulla, sillä emme halunneet käyttöliittymän tilastonäkymään ylimääräistä dataa.

6 Sovelluksen käyttöönotto jatkokehittäjälle

Ohjelmiston lopullinen versio on löydettävissä GitLab sivuston versionhallinnasta osoitteesta https://gitlab.metropolia.fi/tatun/otp-1 .

Ohjelmisto ottaa yhteyden MySQL tietokantaan joka on luotu MariaDB:n avulla. Tietokanta sijaitsee Metropolia AMK:n Educloud -palvelimella. Jotta tietokantaan saa yhteyden tulee yhdistäessä olla Metropolia VPN yhteys päällä. Myös tunnelointi shell.metropolia.fi osoitteeseen vaaditaan.

6.1 Tunnelointi

Tunneloinnin voi suorittaa seuraavilla komennoilla PuTTy -ohjelmiston kautta:

PuTTY: shell.metropolia.fi, port 22
SSH --> Tunnels:

Tietokannalle:
sourceport: "localhost:2206",
destination: "10.114.32.61:3306"

Chatille:
sourceport: "localhost:4998",
destination: "10.114.32.61:4999"

6.2 Hibernate ORM

Hibernate ORM tietokanta yhdistämistyökalulle täytyy myös kertoa kuinka tietokantaan saa yhteyden. Tämän voi tehdä lisäämällä hibernate.cfg.xml nimisen dokumentin peliympäristö/src/main/resources hakemistoon. Dokumentin sisällön tulee näyttää seuraavan laiselta:

```
<hibernate-configuration>
   <session-factory>
      <!-- Use mySQL database -->
```

```
        <property name="hibernate.dialect"> org.hibernate.dialect.MySQL5Dia-
lect</property>
        <!-- Driver, Server, User name, User password -->
        <property name="hibernate.connec-
tion.driver_class">com.mysql.jdbc.Driver</property>
        <property name="hibernate.connection.url">jdbc:mysql://lo-
calhost:2206/gamedatabase</property>
        <property name="hibernate.connection.username">********</prop-
erty>
        <property name="hibernate.connection.password">********</prop-
erty>
        <!-- create , create-drop, update or validate -->
        <!-- Dont use in production -->
        <!-- First time create, when database exists then validate -->
        <property name="hbm2ddl.auto">validate</property>
        <!-- when true, show all SQL-commands in stdout -->
        <property name="show_sql">false</property>
        <!-- model DTO- objects, by packaging -->
        <mapping class="model.Player"/>
        <mapping class="model.PlayedGame"/>
    </session-factory>
</hibernate-configuration>
```

Tähdillä "********" merkityt username ja password tulee vaihtaa oikeaan käyttäjätunnukseen ja salasanaan.

6.3 Vaaditut kirjastot ja tiedostot

Lisää seuraavat tiedostot ja kirjastot Java Build Path:iin:

mysql-connector-java-5.1.49-bin-jar
All required libraries for Hibernate (found in Hibernate-release-version.Final -> lib -> required)
JavaFX (for GUI)
JUnit 5 (for testing)
Maven Dependencies (Project must be maven project)

6.4 Tietokannan luominen

SQL komennot tietokannan kopion luomiseksi löytyvät GitLab versionhallinnan etusivulta. SQL tiedoston nimi on SQL_commands.sql.

6.5 Chat palvelimen luominen

Lähdekoodit chat-palvelin Java-luokille löytyvät myös GitLab versionhallinnasta kansiosta chat_server. Vaihda Socket-parametrit omiksi EchoClient.java ja EchoServer.java-luokissa. EchoClient löytyy peliymparisto/src/main/java/client kansiosta ja EchoServer chat_server/server-kansiosta. Koonnin aikaansaamiseksi aja serveri.

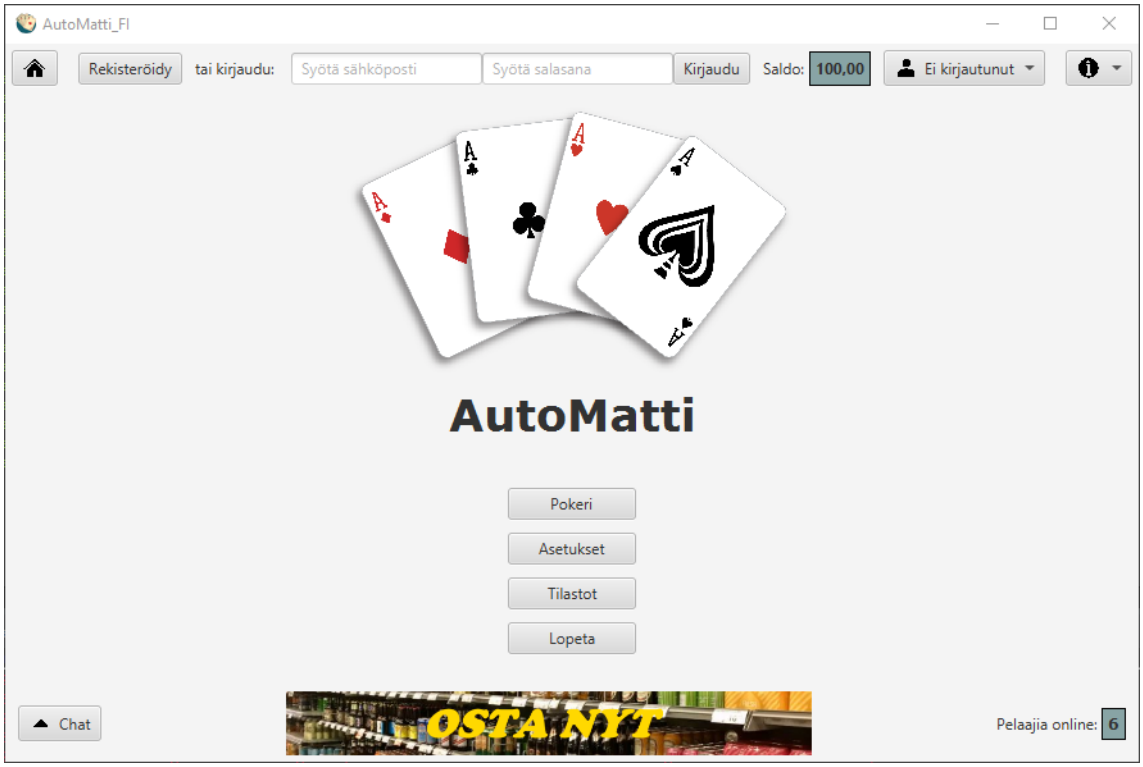
1. Navigoi root-kansioon server-versionhallinnassa.
2. Kirjoita "javac ./server/RunServer.java" ja toista sama kaikille olemassa oleville tiedostoille luodaksesi .class tiedostot.
3. Kirjoita "java server/RunServer" ajaaksesi serveri.

6.6 Helppo asennus

1. Kloonaa GitLabista peliymparisto versiohallinnan sisältö Maven tuettuun projektiin haluamassasi kehitysympäristössä.
2. Aja Main.java luokka, tai View.java

7 Ohjelmiston käyttöohje

Ohjelmisto aukeaa päänäkymään (Kuvio 12), josta löytyy painikkeet muihin osioihin siirtymiselle. Sovelluksen sisältö jakautuu kahteen osaan; staattiseen yläpalkkiin, josta löytyy pikapainikkeet keskeisimmille toiminnoille sekä sen alapuolella olevaan tilaan, jonka sisältö muuttuu sen mukaisesti missä osassa ohjelmistoa ollaan.

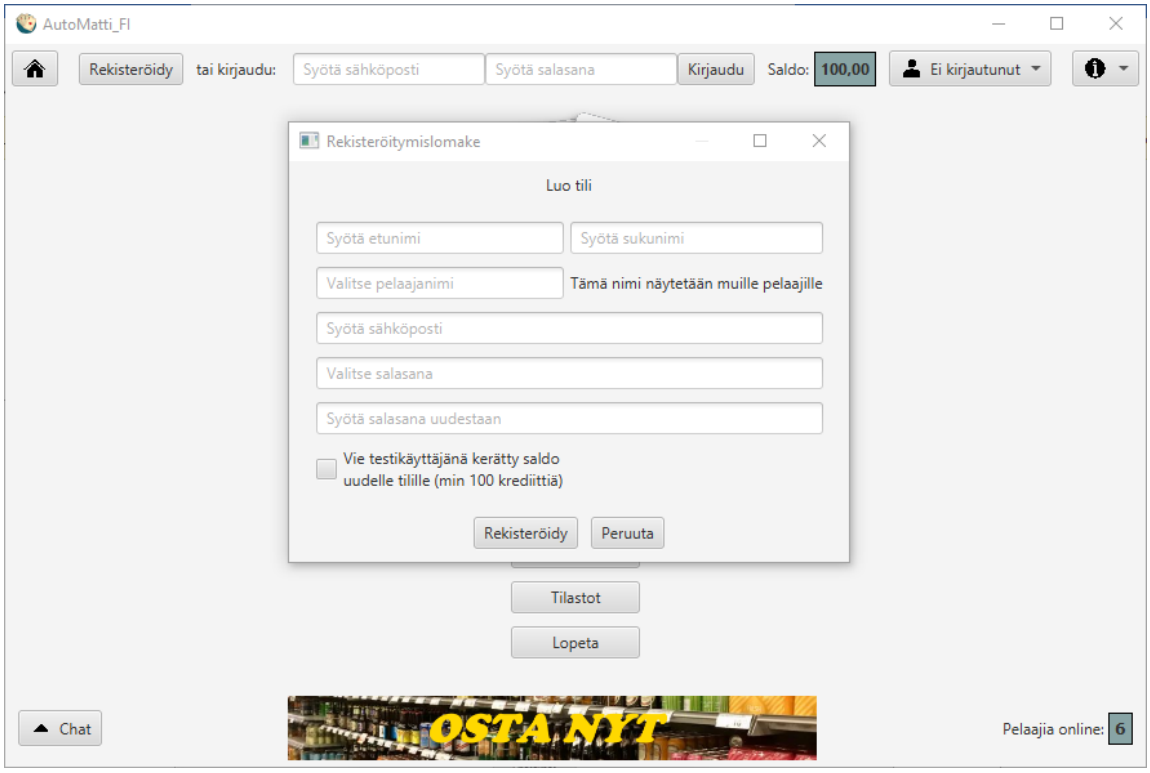


Kuvio 12. Päänäkymä

Yläpalkin oikeassa yläkulmassa on Info-nappi, josta aukeaa napit ohjelman lopettamiselle ja ohjelmiston lisätiedoille. Lopettamispainike toimii mutta lisätieto ikkuna on sisällöltään vielä varsin tynkä. Yläpalkin valittavissa olevat toiminnot vaihtuvat hieman sen mukaan onko käyttäjä kirjautunut vai ei.

7.1 Rekisteröityminen ja kirjautuminen

Saadakseen pelihistoriansa tallentumaan, käyttäjän on rekisteröidyttävä sovellukseen. Rekisteröityminen tapahtuu yläpalkissa sijaitsevasta ”Rekisteröidy”-painikkeesta. Aukeavassa tilin luonti ikkunassa (Kuvio 13) pyydetään syöttämään rekisteröitymiselle olennaiset tiedot, kuten sähköposti ja salasana. Rekisteröityessä annetaan myös mahdollisuus tuoda ilman kirjautumista kerätyt pelimerkit uudelle luotavalle tilille. Tämä on mahdollista vain jos kerättyjä krediittejä on yli 100, sillä minimi aloitussaldo on 100 krediittiä. Rekisteröityminen huomauttaa virheellisistä syötteistä kuten puuttuvista tiedoista ja jo olemassa olevasta sähköpostiosoitteesta.

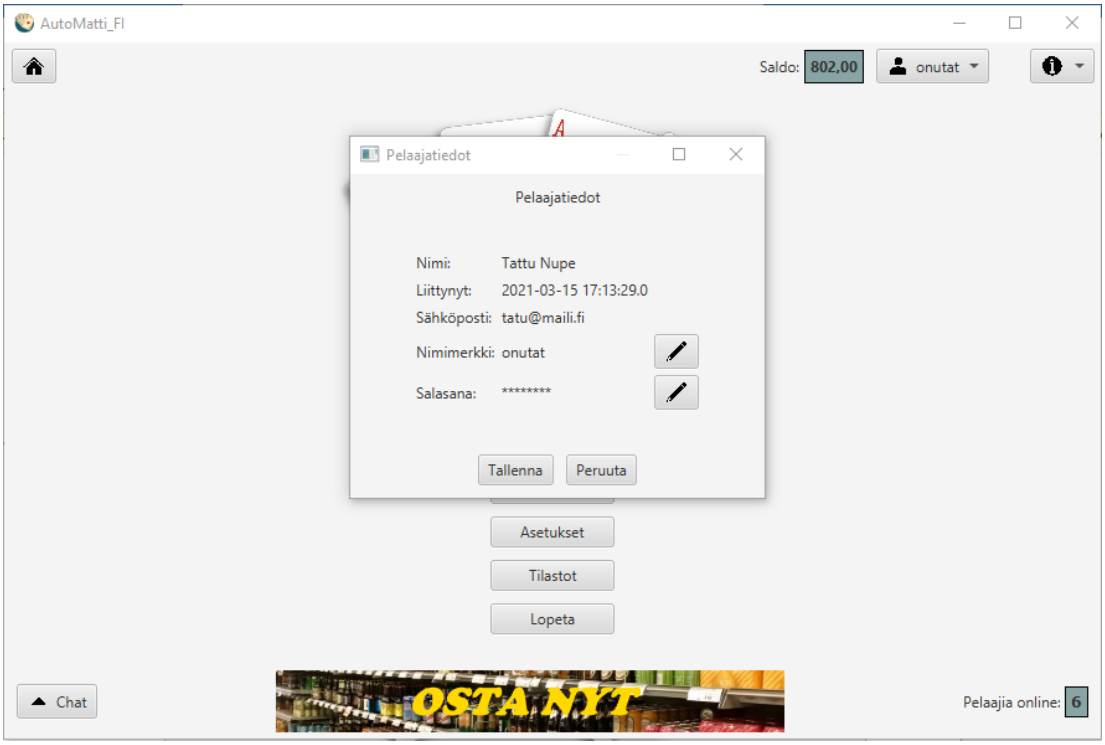


Kuvio 13. Tilinluonti ikkuna

Kirjautuminen tapahtuu myös yläpalkista, siihen vaaditaan rekisteröityessä syötetty sähköposti ja salasana. Tilin luonti kirjaa käyttäjän automaattisesti sisään.

7.2 Pelaajatietojen tarkastelu ja muuttaminen

Kirjautumisen jälkeen käyttäjän pelaajanimi ilmestyy näkyviin "Ei kirjautunut" painikkeen paikalle. Tästä klikkaamalla käyttäjä voi avata pelaajatiedot ikkunan (Kuvio 14) tai kirjautua ulos. Pelaajatiedoissa käyttäjä näkee oman nimensä, aloituspäivämäärän, sähköpostin, pelaajanimimerkin ja tähdillä esitetyn salasanan.

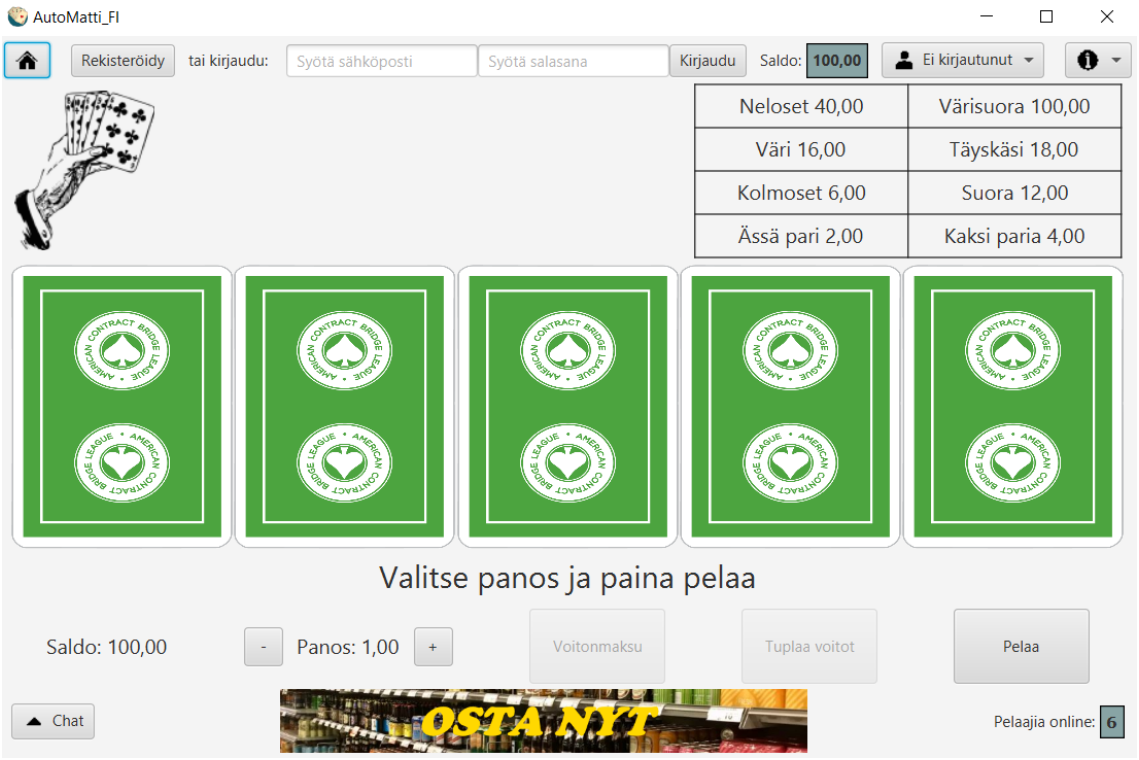


Kuvio 14. Pelaajatiedot ikkuna

Pelaajanimimerkkiä ja salasanaa on mahdollista muokata painamalla muokkaa –nappia (kynän kuva). Tämä tuo näkyviin tekstinsyöttökentän ja vihreän vahvista- sekä punaisen peruuta-napin. Vahvistuksen jälkeenkin muutokset voi peruuttaa, jos ei paina Tallenna painiketta ja poistuu joko Peruuta painikkeella tai sulkemalla pelaajatiedot ikkunan.

7.3 Pokerin pelaaminen

Sovelluksen ensimmäistä ja toistaiseksi ainutta peliä pääsee pelaamaan päänäkymässä (Kuvio 12) olevaa ”Pokeri” -painiketta klikkaamalla, jolloin pokeripelinäkymä (Kuvio 15) aukeaa.



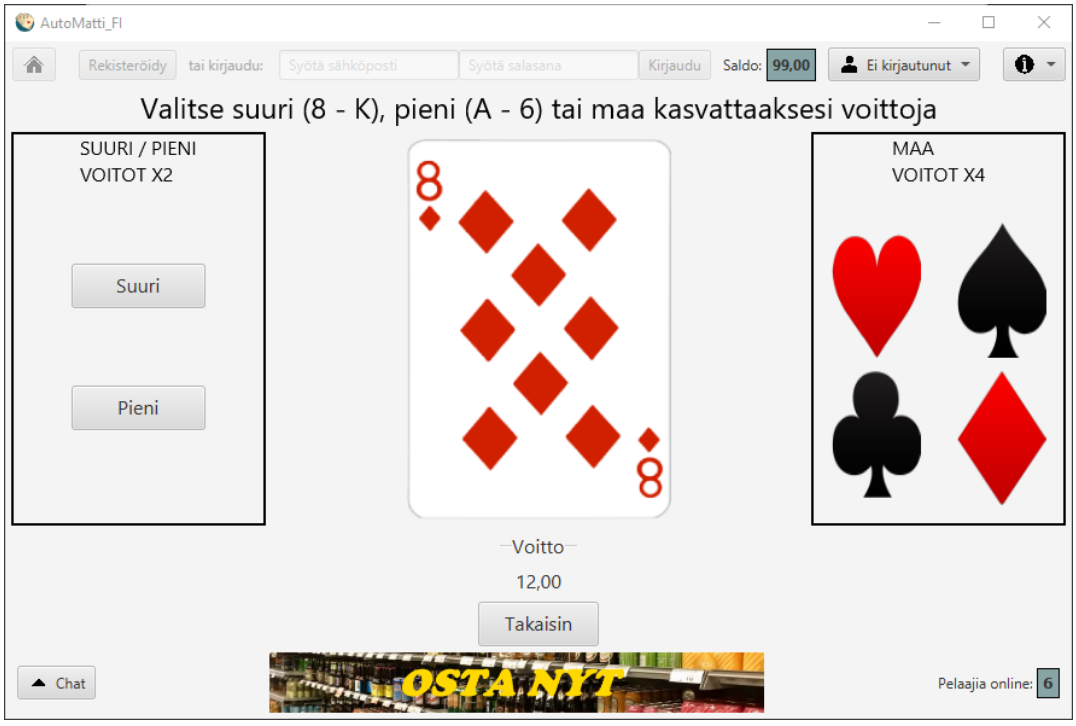
Kuvio 15. Pokeripeli-näkymä

Pelin toiminnallisuus vastaa automaattipokerin toiminnallisuutta, jolle ohjelmasta löytyy painikkeet. Panosta voi muuttaa keskellä alhaalla näkyvillä plus- ja miinuspainikkeilla. Suurin valittava panos on 10 pelimerkkiä ja pienin 0,1. Pelaajan saldon päätyminen alle 0 on estetty. Jos näin käy, käyttäjälle tarjotaan mahdollisuus lunastaa lisää saldoa.

Klikatessa ”Pelaa” ensimmäisen kerran pelaajalle jaetaan kortit. Pelaajan tulee valita klikkaamalla ne kortit, jotka haluaa lukittuvan (Kuvio 16) ja painaa uudelleen oikean alakulman painiketta.

Kuvio 16. Pokeripeli-näkymä, lukitseminen

Jos pelaaja saa voittavan käden, on pelaajalla mahdollisuus voitonmaksua painamalla joko lunastaa voitot, tai yrittää lisätä voittoja valitsemalla tuplaa voitot, jolloin siirrytään tuplausnäkymään (Kuvio 17).

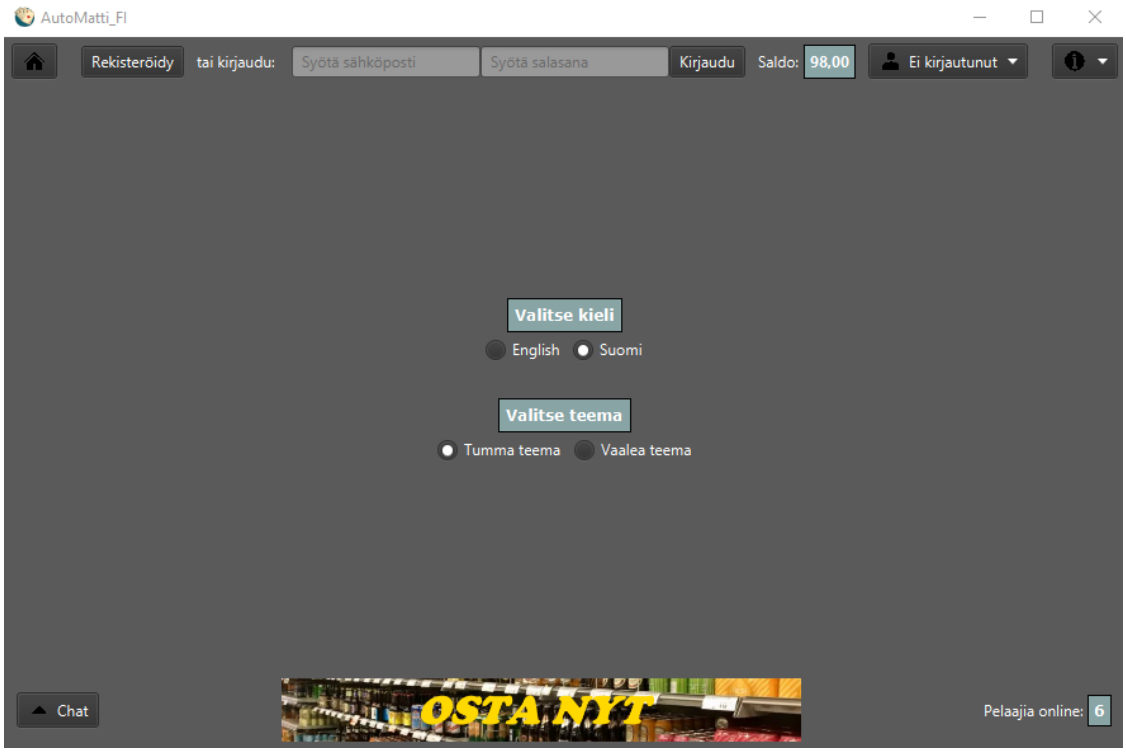


Kuvio 17. Tuplausnäkymä

Tuplauksessa pelaaja voi yrittää kasvattaa alkuperäistä voittoaan valitsemalla pienen (A-6), suuren (8-K) tai maan. Jos pelaaja arvaa oikein, voitot kasvavat. Pelaaja voi jatkaa tuplausta väärään arvaukseen asti, jolloin alkuperäinen voitto hävitään ja pelaaja menettää panoksen verran pelimerkkejä, tai lunastaa voittonsa oikean tuplauksen jälkeen.

7.4 Asetukset

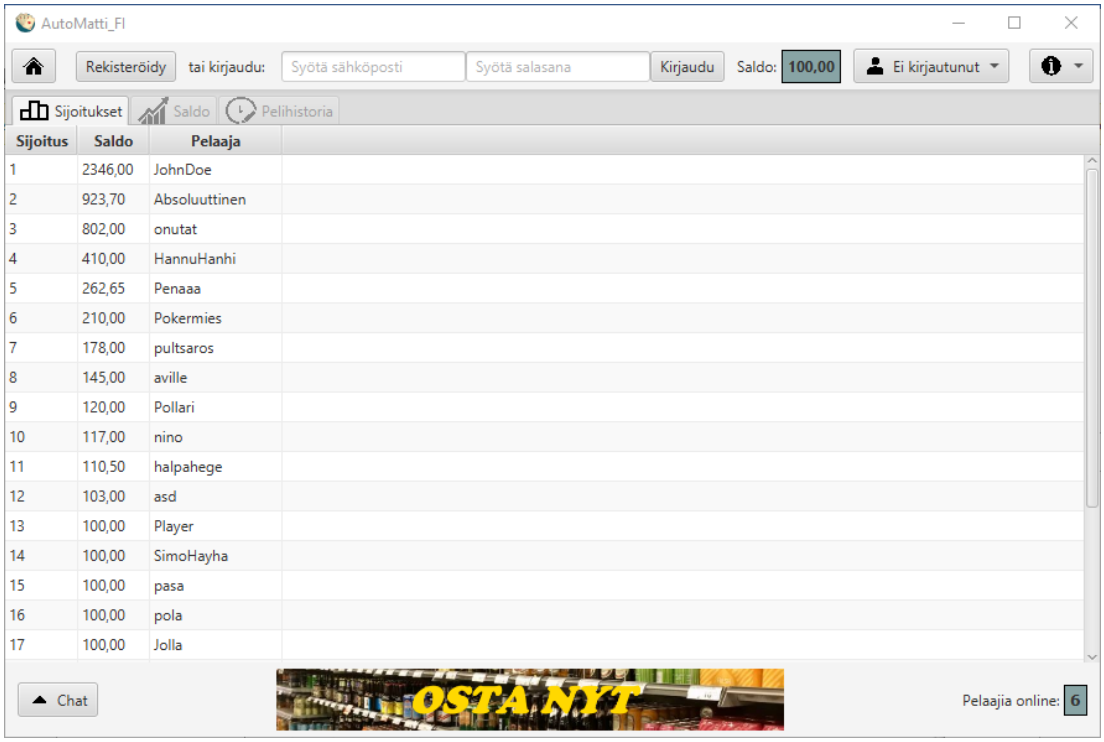
Asetuksista voi vaihtaa kieltä sekä teemaa tumman ja vaalean välillä (Kuvio 18), sekä sovelluksen kieltä. Nämä muutokset tulevat voimaan kaikkialla sovelluksessa.



Kuvio 18. Asetukset-näkymä

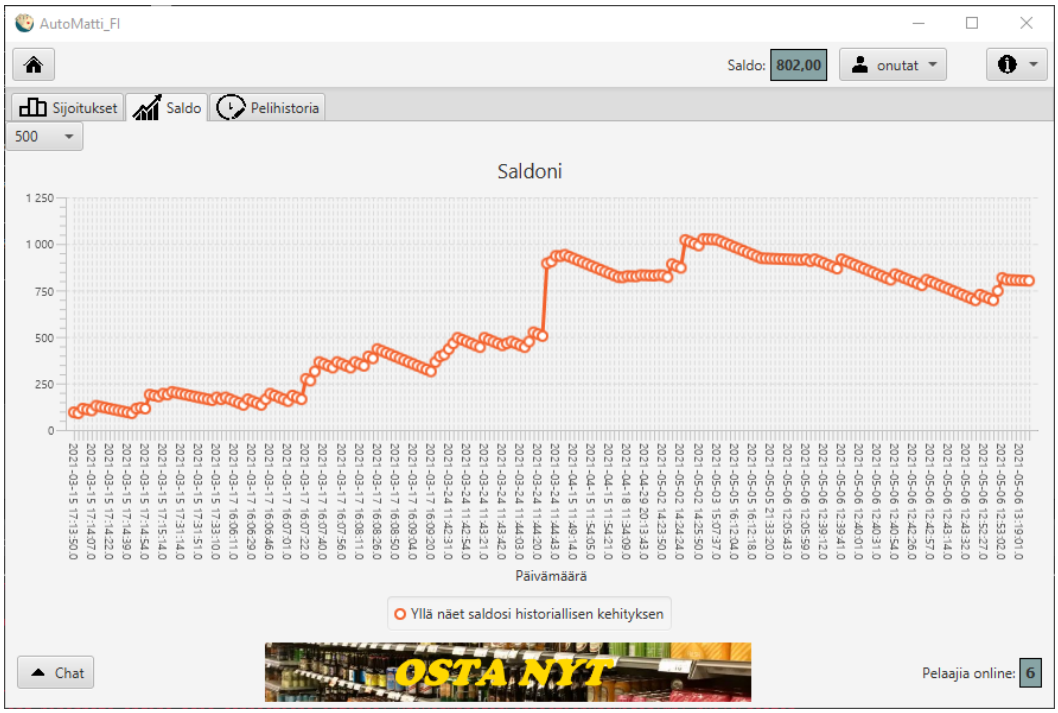
7.5 Tilastot

Tilastot-näkymässä on kolme välilehteä; sijoitukset, saldo ja pelihistoria. Mikäli käyttäjä ei ole kirjautunut sisälle, vain sijoitukset-välilehti on saatavilla (Kuvio 19). Sijoitukset-välilehdessä ilmenee kaikkien rekisteröityneiden pelaajien sijoitus, saldo suuruusjärjestyksessä sekä pelaajan pelaajatunnus.



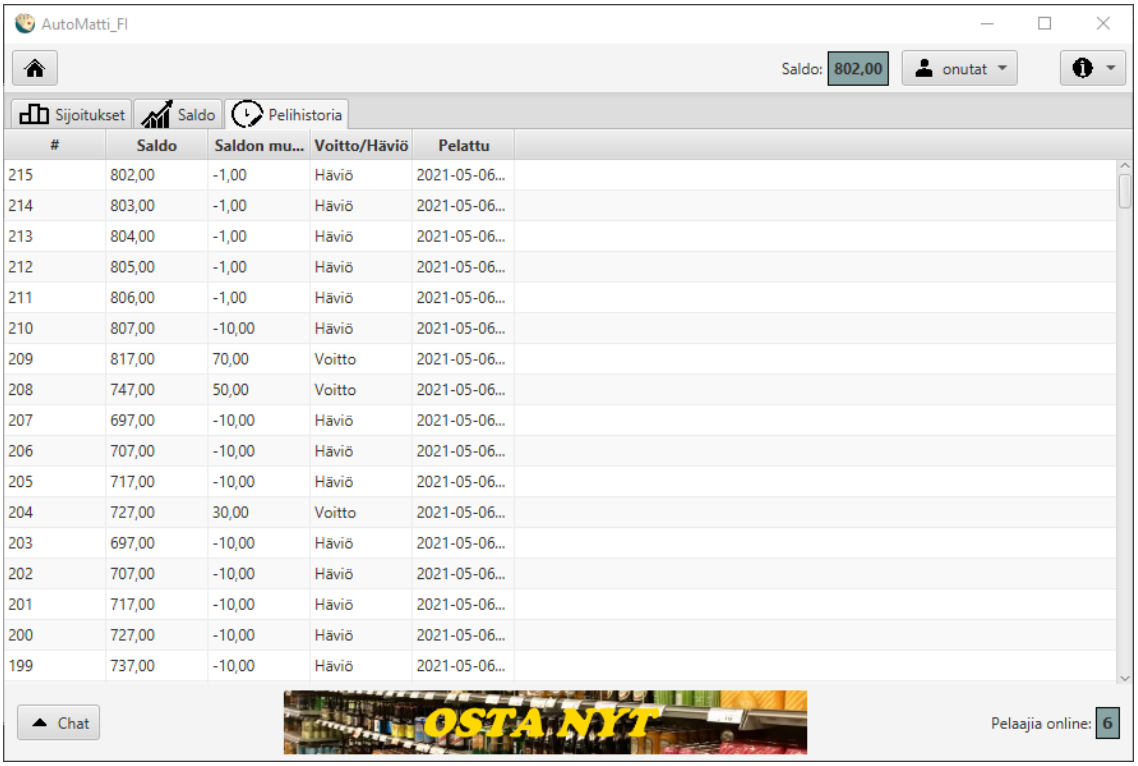
Kuvio 19. Tilastot-näkymä, sijoitukset välilehti

Kun käyttäjä on kirjautunut, Saldo-välilehden viivadiagrammi (Kuvio 20) kertoo oman saldon kehityksen. Käyttäjä voi valita haluaako hän nähdä koko historiakehityksen, viimeisen 50 vai viimeisen 10 pelin kehityksen.



Kuvio 20. Tilastot-näkymä, saldo välilehti

Pelihistoria-välilehdessä (Kuvio 21) listataan kirjautuneen käyttäjän koko pelihistoria alusta lähtien. Lista on käänteinen, jotta viimeisimmät pelit on helppo tarkistaa.



Kuvio 21. Tilastot-näkymä, pelihistoria välilehti

Jokaisen välilehden rivien järjestystä pystyy muuttamaan valitsemalla kolumnien avulla haluttu järjestys. Tämän lisäksi kolumnien järjestystä pystyy muuttamaan raahaamalla haluttu kolumni halutulle paikalle.

8 Yhteenveto

OTP-1 & OTP-2 kurssien lopputuotteena saimme valmiiksi ohjelmiston, joihin voimme olla varsin tyytyväisiä. Projektin aikana opimme paljon uutta muun muassa ketterästä kehitysmenetelmästä ja tiimityöskentelystä, sekä syvensimme Java -osaamistamme. Perusteellisen dokumentoinnin myötä uskomme, että sovelluksen jatkokehittämiseen on hyvät edellytykset, vaikka tiimin rakenne muuttuisi.