

Designdokument

VT-15 DA336A, Grupp 23

Kalle Bornemark 861209

Emil Sandgren 941224

Jimmy Maksymiw 891031

Erik Sandgren 941224

**11/5 2015 V 4.0**

Innehållsförteckning

[Revisionshistorik 1](#_Toc419727920)

[Kontextdiagram 2](#_Toc419727921)

[Beskrivning 2](#_Toc419727922)

[Systemdiagram 3](#_Toc419727923)

[Beskrivning 3](#_Toc419727924)

[Use-case-diagram 4](#_Toc419727925)

[Beskrivning 4](#_Toc419727926)

[Skisser 5](#_Toc419727927)

[Beskrivning 5](#_Toc419727928)

[Klassdiagram 6](#_Toc419727929)

[Klient 6](#_Toc419727930)

[Fragments 7](#_Toc419727931)

[Server 8](#_Toc419727932)

[ER-diagram 9](#_Toc419727934)

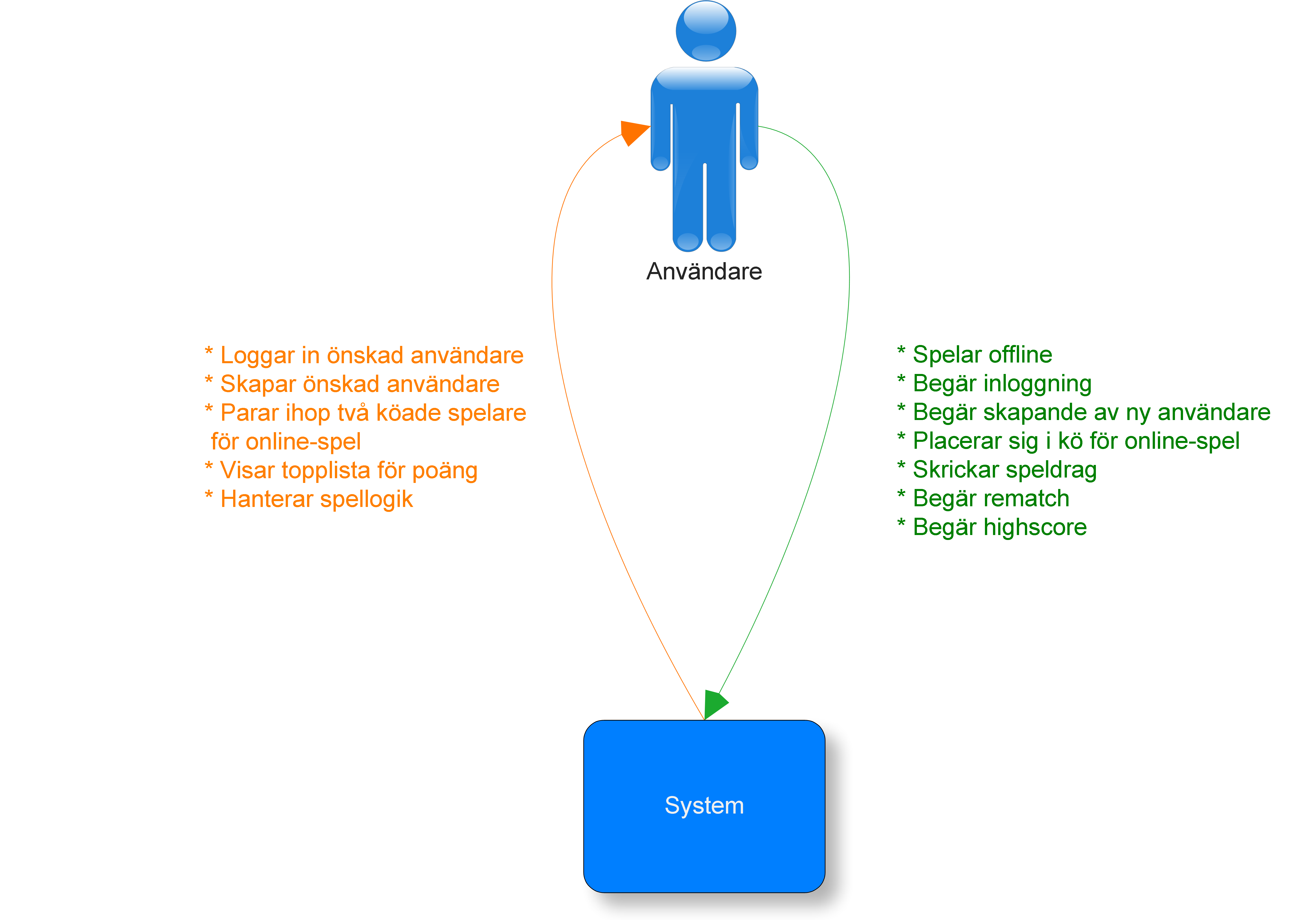
[Beskrivning 9](#_Toc419727935)

# Revisionshistorik

|  |  |  |  |
| --- | --- | --- | --- |
| **Namn** | **Datum** | **Beskrivning** | **Version** |
| Kalle Bornemark | 20/3 2015 | Första utgåvan | 0.1 |
| Erik Sandgren | 22/3 2015 | Lade till systemdiagram | 0.2 |
| Kalle Bornemark | 30/3 2015 | Finslipning till v1.0 | 1.0 |
| Jimmy Maksymiw | 1/4 2015 | Uppdaterat klassdiagram för servern | 1.0.1 |
| Jimmy Maksymiw | 9/4 2015 | Nytt ER-diagram | 1.1 |
| Erik Sandgren | 13/4 2015 | Uppdatering inför v2.0 | 2.0 |
| Erik Sandgren | 24/4 2015 | Uppdaterat klassdiagram och UI-design | 3.0 |
| Jimmy Maksymiw | 11/5 2015 | Uppdaterat ER-diagram | 3.3 |
| Erik Sandgren | 11/5 2015 | Nytt klassdiagram klient/server/Fragment | 3.5 |
| Jimmy Maksymiw | 11/5 2015 | Uppdaterad info om de olika diagrammen. | 4.0 |
| Kalle Bornemark | 18/5 2015 | Lagt till nya diagram och skisser. Uppdaterat alla texter. | 4.1 |

# Kontextdiagram

## Beskrivning

Kontextdiagrammet nedan visar hur användare interagerar med systemet.

# Systemdiagram

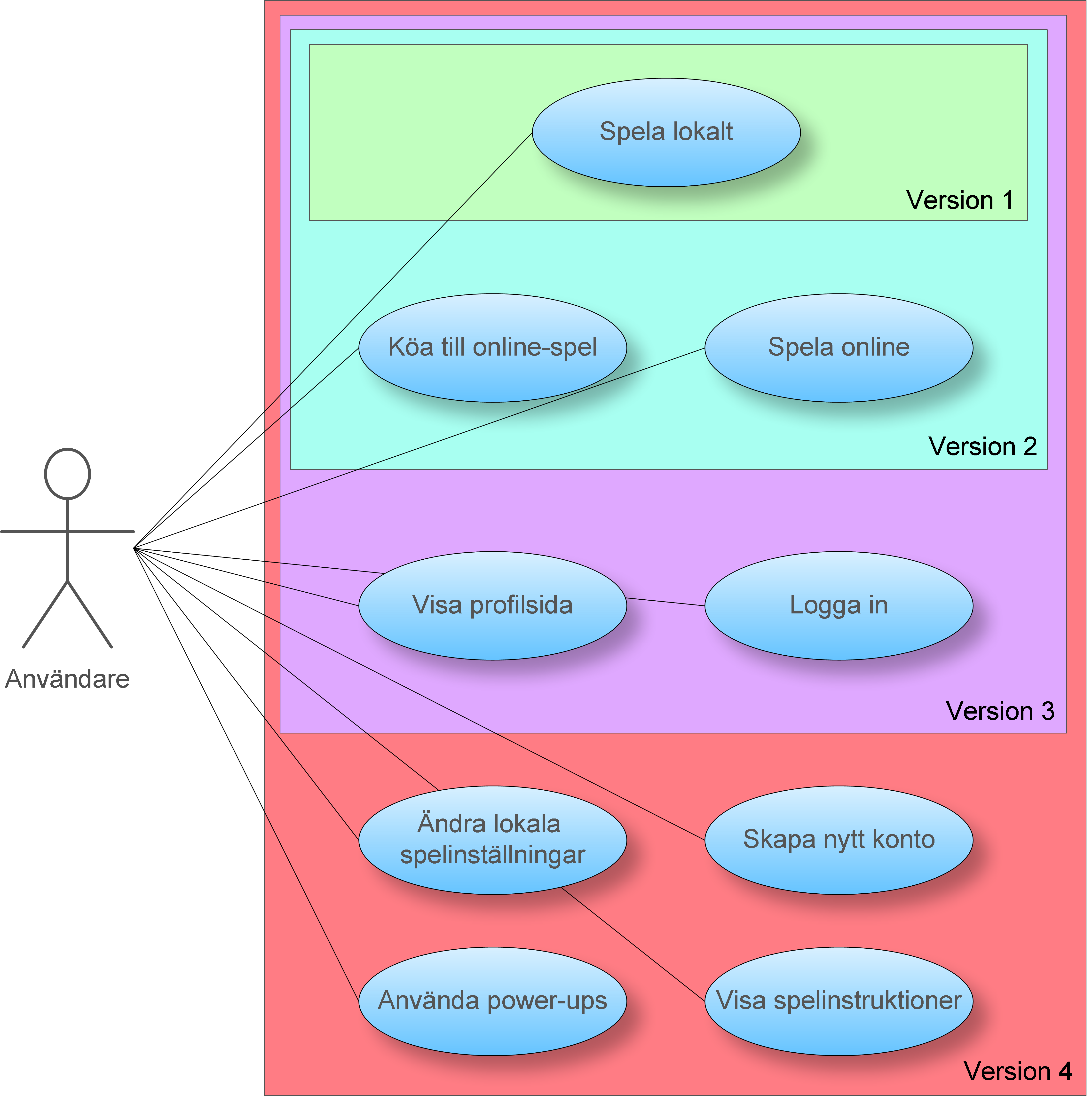
## Beskrivning

Vi använder oss av ett klient/server/databas-system där varje klient som ansluter sig till servern och därmed får tillgång till sitt konto via databasen.

Figuren nedan visar det grundläggande systemupplägget.

# Use-case-diagram

## Beskrivning

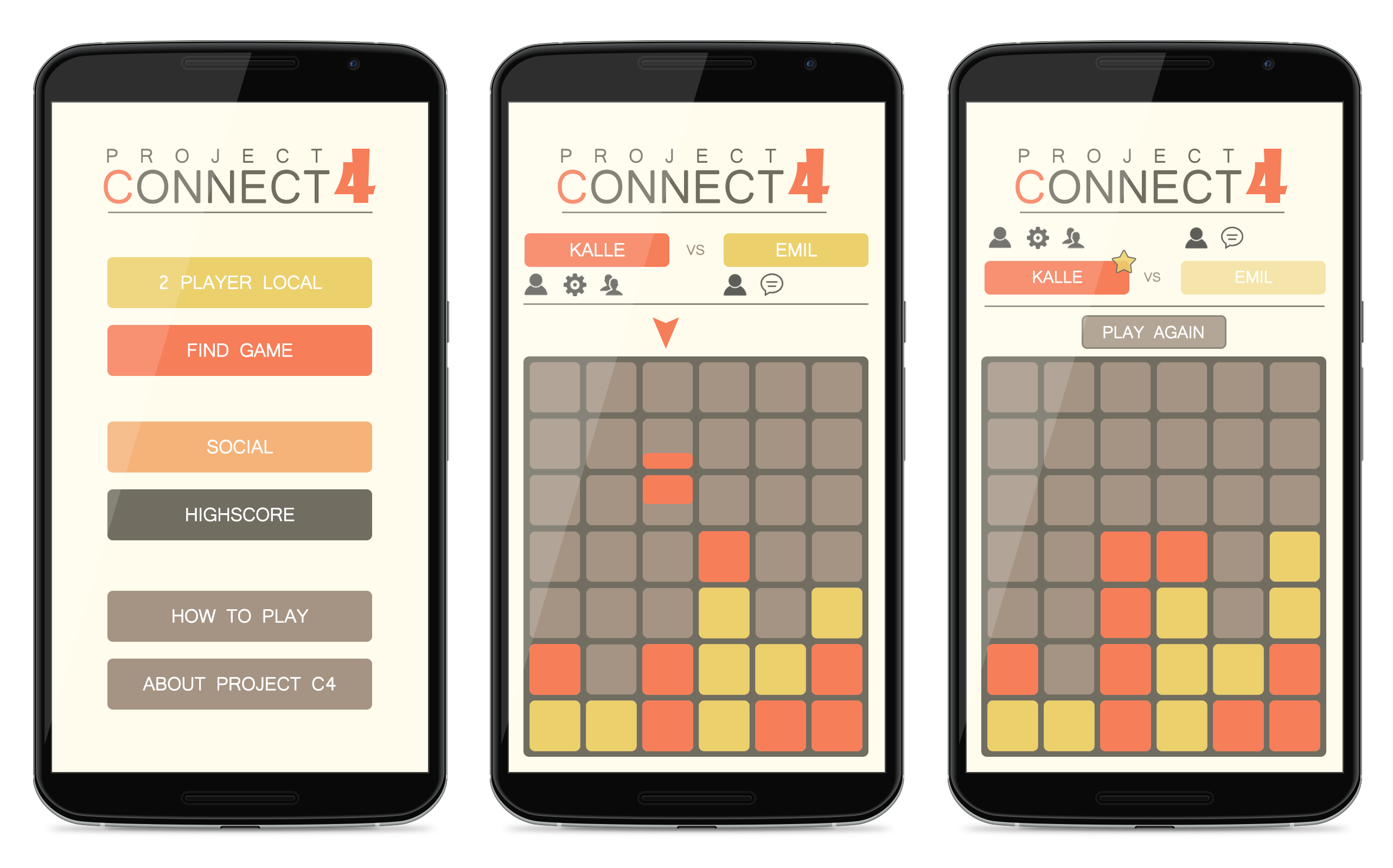
Use case-diagrammet nedan visar vilken funktionalitet användaren har tillgång till i applikationens olika versioner. Detta speglar även den inkrementella utvecklingsprocess vi beskrivit i projektplanen.

# Skisser

## Beskrivning

Skissen nedan visar den grafiska profil vi skapade i förberedelse för utvecklingen av spelet.

En färgpalett togs fram och användes sedan kontinuerligt i alla designelement för att ge ett genomtänkt och enhetligt intryck.



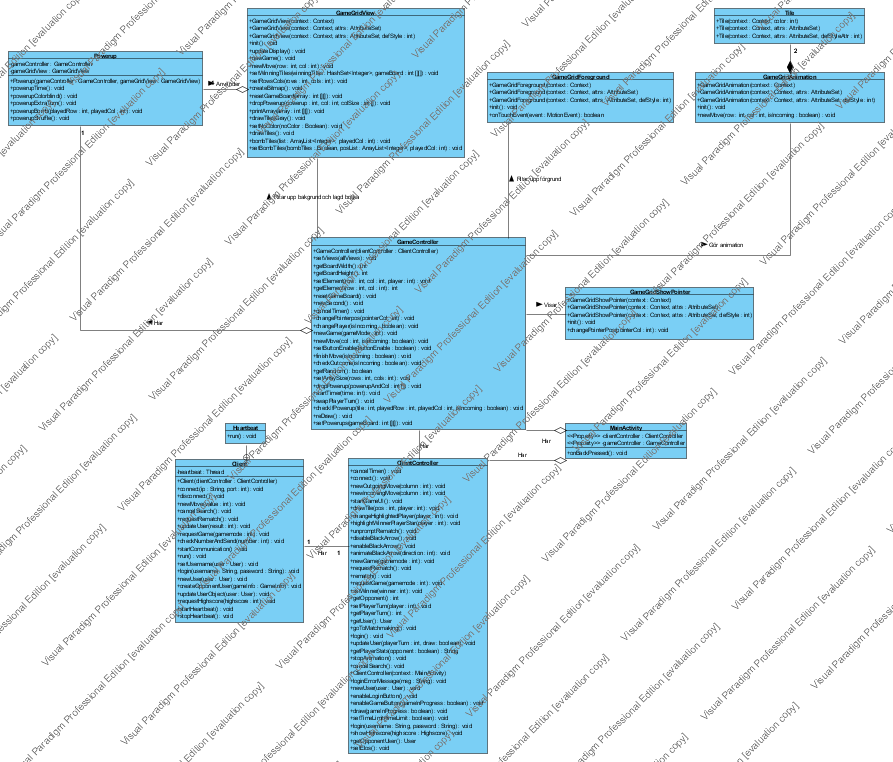
# Klassdiagram

## Klient

Klassdiagrammet nedan visar strukturen för de klasser som hanterar logiken och de som sköter grafiken för spelbrädet.

* Klassen *ClientController* sköter främst logiken mellan klasserna *Client* och *GameController*.
* Klassen *GameControllern* hanterar all logik av spelet och grafiken av spelbrädet.
* Klassen *Client* hanterar kommunikationen mellan applikationen och servern.
* Den inre klassen *Heartbeat* ger en säkrare server/klient-anslutning med felhantering.

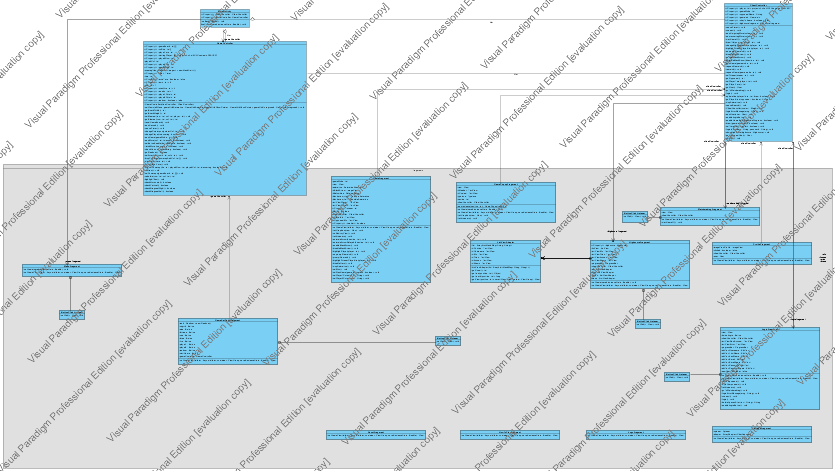
De övriga grafiska komponenterna och deras klasstruktur finns under rubriken Fragments på nästkommande sida.



## Fragments

Klassdiagrammet nedan visar hur de grafiska komponenterna kommunicerar med varandra.

* Det är klassen *ClientController* som hanterar vilket fragment som ska visas beroende på användarens interaktion.
* Om det finns ett pågående spel så sköts grafiken genom klassen *GameController*.

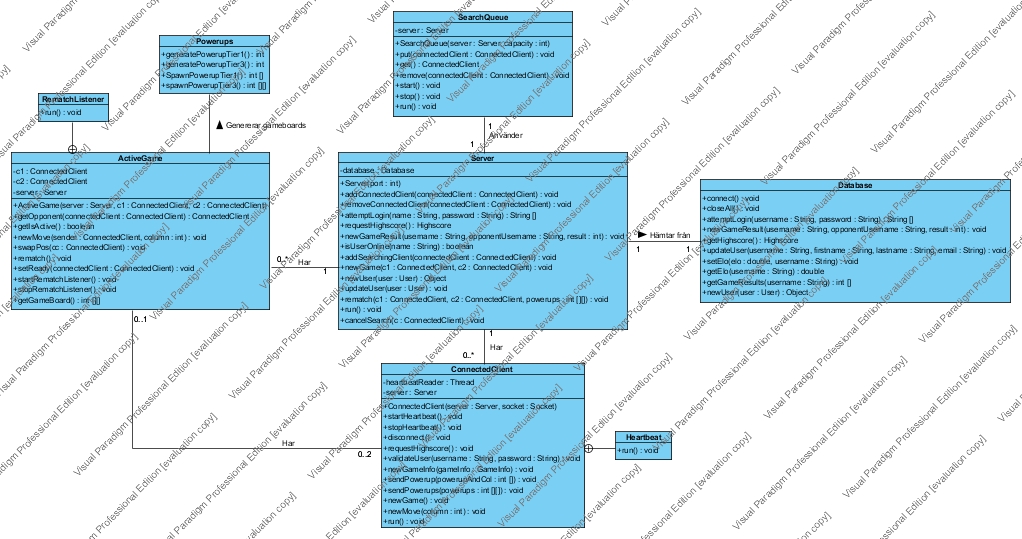


## Server

Klassdiagrammet nedan visar strukturen och kommunikationen mellan klasserna på serversidan.

* Klassen *Server* sköter all kommunikation mellan klasserna *Database* och *ConnectedClient* och har även koll på vilka användare som är anslutna.
* Klassen *ConnectedClient* fungerar som mellanhand mellan klienten och servern, och är den koppling som finns mellan två klienter då de spelar nätverksspel mot varandra.
* Klassen *Database* används för att komma åt den information som ligger lagrad i databasen.
* Klassen *SearchQueue* hanterar de klienter som söker en motspelare att ansluta sig mot för nätverksspel.
* Klassen *ActiveGame* hanterar ett pågående spel mellan två klienter, och använder sig av klassen *Powerups* för att framkalla power-ups på spelbrädet.
* Klassen *RematchListener* används om klienterna vill spela mot varandra igen efter ett avslutat nätverksspel.

## 

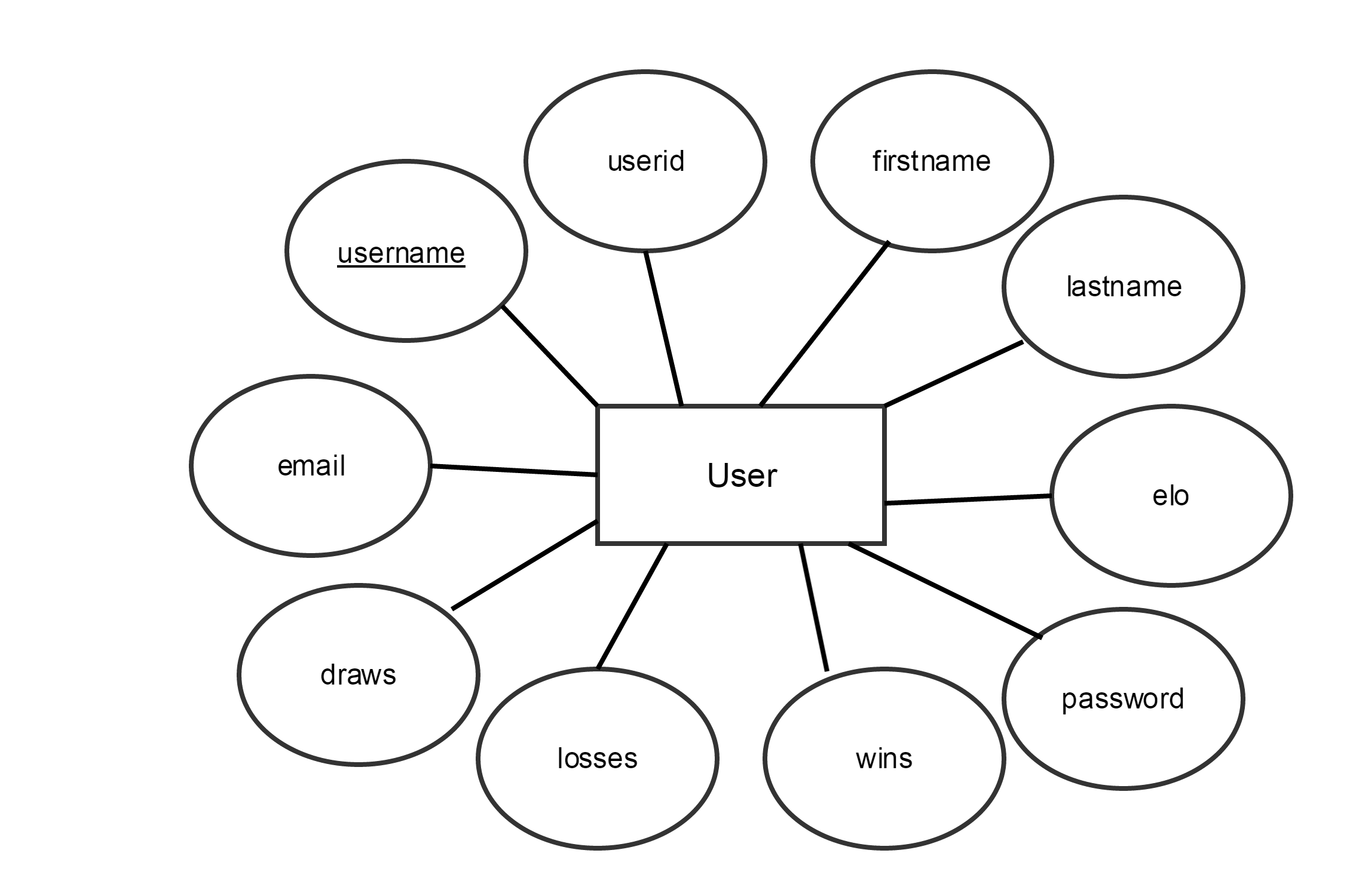


# ER-diagram

## Beskrivning

ER-diagrammet nedan visar vilken information vi lagrar för varje unik användare. Databasen innehåller endast en tabell då användarinformation är allt vi behöver spara.

* Kolumnen *username* används som primärnyckel.



* Kolumnerna *username* och *email* måste vara unika för varje användare.