**Documentatie project Steam**

**Groepsnaam: pay2win**

**Groepsdeelnemers: Thijn, Dieu, Daan, Yverdon, Glenn**

**Inleverdatum: 31 januari 2024**

**Klas: 1vo**

**Inhoudsopgave**

[**Inleiding** 3](#_Toc157463611)

[**Verplichte onderdelen** 4](#_Toc157463612)

[**Basis applicatie met data:** 4](#_Toc157463613)

[**Customer Journey Canvas:** 4](#_Toc157463614)

[**Statestiekalgoritme:** 4](#_Toc157463615)

[**Toepassingen hardware:** 4](#_Toc157463616)

[**Aanvullende onderdelen** 5](#_Toc157463617)

[**Artificial intelligence:** 5](#_Toc157463618)

[**Steam API:** 5](#_Toc157463619)

[**Sorteren:** 5](#_Toc157463620)

[**Zoeken:** 5](#_Toc157463621)

[**Grafieken en diagrammen:** 5](#_Toc157463622)

[**Normaalverdeling:** 5](#_Toc157463623)

[**Technische Informatica:** 5](#_Toc157463624)

[**Extra onderdelen** 6](#_Toc157463625)

[**Library page:** 6](#_Toc157463626)

[**Detail pagina:** 6](#_Toc157463627)

[**Aanmaken van database:** 6](#_Toc157463628)

[**Scripts schrijven voor importeren van data uit API:** 6](#_Toc157463629)

[**Use case diagram:** 6](#_Toc157463630)

# **Inleiding**

Project Steam dient is een applicatie die dient als een ondersteuning bij de data van Steam. Steam is een game platform waar je zowel games kan kopen als verbinding kan zoeken bij andere spelers en dan ook op de community pagina kan posten van het assortiment games dat Steam aanbiedt.

De toegevoegde waarde van onze applicatie is dat je al snel inzage hebt op verschillende aspecten van Steam. De nadruk ligt bij de data van de speler, dit doe je door in te loggen met je Steam account. Je kan dan bijvoorbeeld zien welke games populair zijn bij jouw vrienden, u zult later in het bestand meer lezen over de functionaliteit van de applicatie.

Het document is opgedeeld in drie onderdelen: Verplichte onderdelen, Aanvullende onderdelen en extra onderdelen. Bij deze onderdelen zijn weer verschillende functionaliteiten en deze zal ik ook toelichten bij het desbetreffende kopje.

# **Verplichte onderdelen**

## **Basis applicatie met data:**

De basis applicatie is gemaakt doormiddel van Flask. Dit is een framework waarmee je html kan runnen met Python. De data wordt binnen gehaald doormiddel van requests naar de Steam API. Dit komt terug in een Json formaat en dit wordt gebruikt om de data weer te geven.

## **Customer Journey Canvas:**

Wij hebben dit voor twee momenten gedaan de Ist(het heden), dit is het moment voordat onze applicatie in is gezet en de Soll(de toekomst), dit is het moment nadat onze applicatie in is gezet. Je kan de verbetering zien in de Soll situatie door middel van de verbeterde gedachten van de persona’s. Wij hebben verschillende mensen geïnterviewd en daar mee twee persona’s gecreëerd.

## **Statestiekalgoritme:**

Voor de voorspellende statistieken werd er voorspeld welke games populair gaan worden en welke games er gaan dalen in populariteit. Dit is gedaan met lineaire regressie en wordt gevisualiseerd door middel van een grafiek. Ook de grafiek is een aanvullend onderdeel dus dat wordt later besproken.

De kwalitatieve variabel was de datum in de grafiek voor het voorspellen van de populariteit van games. Deze kan je onder aan de grafiek vinden zodat je weet welk datapunt bij welke datum en tijd hoort.

Voor de kwantitatieve variabel hebben wij de speeltijd van de vrienden gebruikt om de meest populaire games onder je vrienden over de laatste twee weken te bepalen. Dit werd gedaan met behulp van het merge sort algoritme, dit is tevens ook een aanvullend onderwerp, dus hier over meer bij het volgende onderdeel. De data komt binnen via de getPlayerGames API

## **Toepassingen hardware:**

De gekozen toepassing is het laten zien van personastate van jouw vrienden op de neopixel. Als het lampje bijvoorbeeld groen is, is de persoon online en als het rood is, is de persoon offline. Het ophalen van de gegevens gebeurt via de getplayersummaries api.

# **Aanvullende onderdelen**

## **Artificial intelligence:**

### **Steam API:**

De data die in de gui te zien is komt uit verschillende Steam API’s. Als het heel veel data was is de data opgeslagen in de database. Dit hebben wij bijvoorbeeld gedaan met alle games op Steam. Op deze manier hoeven wij niet voor elke game een API-call te make en kunnen wij dit dus uit de database ophalen.

### **Sorteren:**

Zoals eerder benoemd, is er gebruik gemaakt van het merge sort algoritme op de games die populair zijn tussen vrienden. Dit was eerst het insertion sort algoritme, maar dit was niet snel genoeg. Merge sort heeft een voorkeur en aan gezien het ligt aan hoeveel vrienden de gebruiker heeft en hoeveel games zij de afgelopen twee weken hebben gespeeld, is het goed om ervan uit te gaan dat de dataset groot gaat worden.

### **Zoeken:**

Er is een simpel zoek systeem geïmplementeerd die alle games uit de database zoekt dit gebeurt via ajax(Thijn, klopt dit?).

### **Grafieken en diagrammen:**

Zoals eerder benoemd is er een grafiek gebruikt voor het visualiseren van de voorspelling voor de populariteit van de games. Dit gebeurd met JavaScript(wat specifiek met JavaScript?).

### **Normaalverdeling:**

(Hebben wij wat in de applicatie voor de normaalverdeling)

## **Technische Informatica:**

Als ondersteuning voor de Neopixel is er een lcd-scherm. Hierop krijg je te zien welke spelers op welk lampje zitten.

# **Extra onderdelen**

## **Library page:**

Op deze pagina kan je al de games zien die jij in je bezit hebt. De lijst voor de game Ids wordt opgehaald vanuit de api en de games worden opgehaald uit de database. De games komen dan tevoorschijn met het plaatje en de naam van de game. Het is de bedoeling dat je dan op de game kan klikken zodat je doorverwezen kan worden naar de detail pagina.

Gemaakt door: Glenn

## **Detail pagina:**

De detail pagina laadt met de id die mee is gegeven via de routing van de pagina. Als je namelijk op het plaatje van de game op de library pagina klikt dan staat de id van die game meegegeven in de url. Met deze id wordt de game opgehaald uit de database en dingen als de naam, beschrijving, de requirments voor je pc, screenshots, ect.

Gemaakt door: Daan

## **Aanmaken van database:**

Om ervoor te zorgen dat de site niet heel erg traag wordt en er heel veel API requests moeten komen is er een database gemaakt. Eerst was er een ERD opgesteld en daarna is dat omgezet naar de database.

Gemaakt door: Dieu

## **Scripts schrijven voor importeren van data uit API:**

Om de data in de database te krijgen zijn er scripts gemaakt die een API request maakt voor alle game id’s en deze id’s een voor een in een request zet die de details van deze games ophaalt. Dit gaat dan door een functie die filtert of het ook echt van het type “game” is en geen DLC of iets anders, en dan worden alle gegevens verdeeld over de desbetreffende tabellen en kolommen.

Gemaakt door: Thijn

## **Use case diagram:**

Om in kaart te brengen wat wij allemaal gebruiken en waar dit vandaan komt is er een Use Case diagram gemaakt.

Gemaakt door: Dieu