Conversphere

1st Feil Lukas
OTH Amberg-Weiden
Bad Kötzting, Deutschland
l.feil@oth-aw.de

2nd Markus Fleischmann
OTH Amberg-Weiden
Wernberg-Köblitz, Deutschland
m.fleischmann2@oth-aw.de

3rd Stefan Reger
OTH Amberg-Weiden
Schwarzenfeld, Deutschland
s.reger@oth-aw.de

4th Lukas Rupp
OTH Amberg-Weiden
Amberg, Deutschland
l.rupp@oth-aw.de

5th Sorel Tahata Djoumsi
OTH Amberg-Weiden
Regensburg, Deutschland
s.tahata-djoumsi@oth-aw.de

***Abstract*—Chat Wep App mit Raumfunktion und Proximity-Chat**

***Index Terms*—PWA, Chat, Proximity, Room, Web App, Angular, NodeJS, MongoDB, Express**

I. Einführung

Conversphere ist eine Web-Chat-Anwendung, die es Benutzern ermöglicht, miteinander zu kommunizieren. Die Kommunikation zwischen den Usern erfolgt über Textnachrichten. Es gibt eine Karte, auf welcher die eigene Position auswählbar ist und die Positionen der anderen User anzeigt. Es können nur die Nachrichten von Usern empfangen werden, wenn man innerhalb eines gewissen Umkreises dieser User positioniert ist.

Die hörbaren Nachrichten sind in einem Chat-Feld sichtbar. Jede Nachricht ist mit dem Namen des Absenders gekennzeichnet, so dass Nachrichten den verschiedenen Usern zugeordnet werden können. Der Benutzer kann eine Nachricht über ein Textfeld im Chat-Feld senden.

Der Benutzer kann seine Lautstärke der Nachricht wählen (flüstern, normales Gespräch, oder schreien). Dies beeinflusst die Reichweite der Nachricht. Ebenso sind Spezialeffekte wählbar (betrunken, Rage-Mode, usw.), was Qualität der Nachricht beeinflusst. Ein User hat die Möglichkeit, sich ein Profil auf der Seite zu erstellen, um seinen Charakter zu personalisieren. Es können neue Chat-Rooms erstellt werden und man kann schon existierenden Chat-Rooms über deren ID beitreten.

II. Motivation

Unsere Motivation war es, eine neue Art des Austausches mit unserem einzigartigen Chatroom zu erschaffen. Durch unsere bisherigen Erfahrungen mit den vorhandenen Chats, stießen wir immer wieder auf Langeweile und Unzufriedenheit. Damit waren wir auch nicht allein.

Unser Chatraum soll anders sein als alles, was du bisher erlebt hast.

Durch den individuellen Chat-Radius kann der neuste Klatsch und Tratsch ausgetauscht werden, der vielleicht nicht für jedermann Ohren bestimmt ist. Die Auswahl deiner Chatpartner bietet die perfekte Grundlage dafür. In Zeiten von Remotearbeit immer essenzieller wird, wollten wir zudem einen sicheren Raum schaffen, in dem es möglich ist, seine Gedanken und Ideen teilen zu können und gleichzeitig sozial distanziert zu bleiben.

III. Verwandte Arbeiten

Inspiration für das Projekt boten Websites wie Gather [1] oder das mittlerweile beendete Projekt von Skittish [2].

Diese besitzen einen Proximity-Chat bzw. Näherungschat mit Sprachfunktion. Wenn man mit seiner Spielfigur in einer gewissen Entfernung zu anderen Spielern steht kann mit ihnen reden. Verlässt man diesen Umkreis wieder hört man die anderen Spieler nicht mehr.

Für Pandemie-/Quarantänezeiten, Freunde in einer anderen Stadt oder um neue Leute kennenzulernen ist dies eine gute Möglichkeit zu Kommunizieren.

Conversphere möchte dieses Grundprinzip als Only-Textchat-Platform adaptieren.

IV. Alleinstellungmerkmal

Conversphere will als möglichst simple und intuitive Chat-Plattform dienen. Da das Design sehr schlicht gehalten wird, hebt sich die Anwendungen vom aktuellen Trend ab, bei welchem immer mehr aufwendige Animationen und Werbung fast schon Standard sind.

Bei Conversphere soll der Austausch im Vordergrund stehen. Da die Anwendung nur Chat-basiert ist, kann sich jeder an Gesprächen beteiligen und muss keine Angst haben, aufgrund von Geschlecht oder Alter anders behandelt zu werden.

Durch das Profil ist es möglich, dass zusätzliche Informationen gespeichert und anderen Nutzern zu Verfügungen gestellt wird. Dadurch ist es für jeden Nutzer selbst frei wählbar, welche Informationen er mit anderen Nutzern teilen möchte. Durch den Verzicht auf Sprach- und Videochats kann man auch völlig anonym mit anderen Nutzern chatten.

V. Technisches Grundkonzept

A. Datenbank - Mongo DB

Zur Speicherung von Nutzer und Spiel bzw. Chatdaten soll eine NoSQL MongoDB Datenbank zum Einsatz kommen. MontoDB Atlas bietet dafür einen kostenlosen Service, welcher eine gehostete MongoDB zu Verfügung stellt.

B. Backend - Express.js

Das Backend soll auf einem in Node.js laufenden Express.js Server basieren. Dieser soll die gesammte Spiellogik implementieren und alle anfallenden Daten zur Speicherung in die Datenbank leiten.

C. Schnittstellen - Restful API

Der Express Server sowie das Frontend sollen über eine Restful API miteinander Kommunizieren. Es sollen dabei für jede Funktion bzw. jede Art von Daten eine eigene Domain sowohl im Frontend als auch im Backend erstellt werden.

D. Frontend - Angular PWA

Für das Frontend soll mit Hilfe von Anuglar eine Progressiv Web App erstellt werden welche mit Hilfe einer Render bzw. Animierungsbiliothek erweitert wird.

E. Deployment - AWS S3 und ECS

Die Webseite soll als static Website über AWS S3 an die Clienten verteilt werden. Das Backend soll mit Hilfe eines Docker Containers in der AWS ECS gehostet werden.

VI. Anforderungen

Die Anforderungen werden in Form von User-Storys mit Akzeptanzkriterien formuliert. Die MUSS-Anforderungen definieren das “Minimum Viable Product“ (MVP). Die SOLL-Anforderungen sollen in den nachfolgenden Iterationen fertiggestellt werden, wobei die Userstorys unter OPTIONAL mögliche Erweiterungen für das Produkt bieten, welche je nach Zeitbedarf umgesetzt werden können.

A. MUSS-Anforderungen

1) Raum erstellen:
Ich als Nutzer möchte über einen Knopf einen neuen Raum erstellen können, um dort einen neuen Chat zu starten, welcher am Anfang bis auf mich leer ist.

Akzeptanzkriterien:

- Eingabe des Nicknames, mit welchen man im Raum angezeigt wird.
- Weiterleitung in neuen Raum nach Button-Druck
- Anzeige der Raum-ID für diesen Raum
- Chatroom wird angezeigt

2) Raum beitreten:
Ich als Nutzer möchte nach Erhalt eines Codes durch einen anderen Spieler nach Eingabe des Codes mittels einem Knopfdruck dessen Raum beitreten, um dort an dem Chat teilzunehmen.

Akzeptanzkriterien:

- Eingabe des Nicknames, mit welchen man im Raum angezeigt wird.
- Nach Eingabe der Raum-ID wird bei einem ungültigen Wert eine Meldung an den Nutzer ausgegeben.
- Bei einer gültigen Raum-ID wird man in den Raum weitergeleitet.
- Chatroom wird angezeigt

3) Schreiben von Nachrichten:
Ich als Nutzer möchte über eine Chatbox Nachrichten eingeben können.

Akzeptanzkriterien:

- Die Nachricht wird im Textfeld angezeigt.
- Das Textfeld soll die letzten 10 Nachrichten anzeigen. Mit einem Schieberegler kann die Anzahl der Nachrichten, welche angezeigt werden, verändert werden.

4) Bewegen auf der Karte:
Ich als Nutzer möchte mich auf der Karte bewegen können.

Akzeptanzkriterien:

- Der Spieler kann seine Position mit einem Klick auf der Karte auswählen.
- Die Position soll auf der Karte sichtbar sein.
- Der Spieler kann seine Position jederzeit ändern.
- Der Spieler soll jederzeit seine Position auf der Karten sowie die Position andere Spieler sehen können.

B. SOLL-Anforderungen

1) Lautstärke der Nachricht einstellen:
Ich als Nutzer möchte über einen Schieberegler einstellen können, wie weit man meine Nachrichten lesen kann.

Akzeptanzkriterien:

- Reichweite einstellbar über den Schieberegler.
- Nach verschieben ändert sich der Umkreis.

2) Darstellung der Nachricht:
Ich als Nutzer möchte, dass je nach Lautstärke der erreichten Nachrichten diese in unterschiedlicher Weise angezeigt werden, so dass es leichter erkennbar ist, welche Nachrichten von Spieler im näheren Umkreis stammen.

Akzeptanzkriterien:

- Je nach Entfernung und Lautstärke der Nachricht wird diese mit unterschiedlichem Verblassungs-Effekt angezeigt.

3) Drunk-Mode:
Ich als Nutzer möchte einen Drunk-Mode, in welchem Nachrichten in welchem Buchstaben zufällig vertauscht werden.

Akzeptanzkriterien:

- Aktivierung des Drunk-Modes über einen Button im Chat-Feld.
- Nachrichten im Drunk-Mode werden den mit zufällig vertauschten Buchstaben innerhalb eines Wortes angezeigt.

4) Rage-Mode:
Ich als Nutzer möchte einen Rage-Mode, in ich meine Nachricht an alle Teilnehmer im Raum senden kann.

Akzeptanzkriterien:

- Durch das senden einer Nachricht im CAPS-Lock wird der Rage-Mode aktiviert.
- Die gesendete Nachricht ist für alle Nutzer sichtbar.

C. OPTIONAL-Anforderungen

1) Registrieren:
Ich als nicht registrierter Nutzer möchte eine Möglichkeit besitzen, mich zu registrieren, um mein Profil dauerhaft speicher zu können.

Akzeptanzkriterien:

- Button “Registrieren“ leitet an das Formular mit den Angaben weiter
- Formular frägt folgen Werte ab: E-Mail und Nutzername (müssen eindeutig sein) und Passwort
- Nach Prüfung, ob die E-Mail und der Nutzernamen eindeutig sind, wird eine Bestätigungsmail versendet
- Erst nach Bestätigung wird der Account freigeschaltet

2) Anmelden:
Ich als registrierter Nutzer möchte eine Möglichkeit besitzen, mich anzumelden, um auf meinen Account zuzugreifen.

Akzeptanzkriterien:

- Button “Login“ leitet an ein Formular für die Eingabemöglichkeit der E-Mail/Nutzernamen und des Passworts weiter
- Nach erfolgreicher Eingabe wieder Weiterleitung auf den Startbildschirm, jedoch ist erkennbar, dass man eingeloggt ist

3) Nutzerprofil bearbeiten:
Ich als angemeldeter Nutzer möchte eine Möglichkeit besitzen, dass ich auf meine Profilseite zugreifen kann, und dort Eigenschaften wie Nickname, Uhrzeit/Ort einstellen kann.

Akzeptanzkriterien:

- Button “Profil“ leitet auf die Profilseite weiter
- Eigenschaften werden angezeigt
- Bei Änderungen an den Eigenschaften wird der “Speichern“-Button aktiv
- Nur bei der Betätigung des “Speichern“-Buttons werden die Änderungen gespeichert, ansonsten verworfen
- Wird die Profilseite verlassen, ohne dass Änderungen gespeichert sind, so wird der Nutzer darauf aufmerksam gemacht

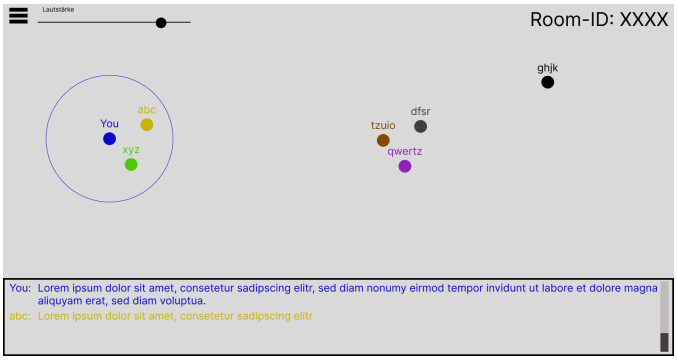

Fig. 1. Skizze eines Chat-Rooms

REFERENCES

[1] https://www.gather.town/
[2] https://skittish.com/