

<input type="checkbox"/>	Gr. 1, M. Winkler, BSc MSc	Name	Jack Heseltine	Aufwand in h	15
<input checked="" type="checkbox"/>	Gr. 2, Dr. E. Pitzer	Punkte	Kurzzeichen Tutor / Übungsleiter _____ / _____		

Beispiel	L Lösungsidee	I Implementierung	T Tests	S = L+I+T	M Multiplikator	S*M
1	□□□	□□□□	□□□		10	
Erfüllungsgrad ankreuzen bzw. Summen eintragen und auch online ausfüllen!						

Wie angekündigt, wollen wir in den restlichen Übungen – also von der vorliegenden 5. bis zur 7. Übung – anhand *einer* Aufgabenstellung verschiedene Möglichkeiten der Java-Klassenbibliothek ausloten. Damit Sie die weiteren Übungen meistern können, sollten Sie von Anfang an mitmachen.

ChatBPT – Chat-Based Platform for Togetherness (Gesamtüberblick & Vorschau aller Ausbaustufen)

Kurzbeschreibung

Im Rahmen dieser Projektarbeit soll eine Plattform entwickelt werden, die es Benutzer*innen ermöglicht, sich in Chat-Räume zusammenzuschließen und in Echtzeit miteinander zu kommunizieren. Die Anwendung unterstützt eine unbegrenzte Anzahl von Chat-Räumen, in denen sich Benutzer*innen zu verschiedenen Themen austauschen können.

Die Chat-Räume sind öffentlich, sodass Benutzer*innen einem Chat-Raum beitreten und die Nachrichten von anderen Benutzer*innen im Chat-Raum sehen können. Es gibt jedoch auch die Möglichkeit, private Nachrichten direkt zwischen Benutzer*innen zu senden, die nicht im Chat-Raum angezeigt werden.

Die Plattform besteht aus einer Chat-Anwendung (*ChatBPT-UI*), welche von den Benutzer*innen zur Interaktion mit anderen Benutzer*innen verwendet wird und einer Server-Anwendung (*ChatBPT-Hub*), die für die Koordination des Nachrichtenaustauschs und die Administration der Benutzer*innen und Chat-Räume verantwortlich ist. Nachfolgend werden die Anforderungen an diese beiden Komponenten im Detail beschrieben.

Chat-Anwendung (ChatBPT-UI)

- *Login*: Beim Starten der Chatanwendung müssen sich Benutzer*innen anmelden. Falls diese noch nicht über ein Konto verfügen, müssen sie sich registrieren.
- *Registrierung von Benutzer*innen*: Benutzer*innen müssen sich unter Angabe eines Namens, eines Kurznamens und eines Passworts registrieren können.
- *Erzeugen eines Chat-Raums*: Alle Benutzer*innen haben die Möglichkeit, Chat-Räume zu erstellen. Ein Chat-Raum ist durch einen Namen (Thema) und seine Mitglieder gekennzeichnet. Der Ersteller eines Chat-Raums hat spezielle Rechte, die im Folgenden genauer definiert sind.
- *Beitritt zu einem Chat-Raum*: Benutzer*innen können einem Chat-Raum beitreten. Dafür ist nur die Kenntnis des Namens des Chat-Raums erforderlich.
- *Verbannen von Benutzer*innen*: Der Erzeuger eines Chat-Raums kann Benutzer*innen aus dem Raum verbannen, was zur Folge hat, dass diese keine Nachrichten mehr an den Raum senden können und auch keine Nachrichten erhalten. Dieser Bann kann vom Erzeuger auch wieder aufgehoben werden.

- *Löschen eines Chat-Raums:* Der Erzeuger eines Chat-Raums kann diesen auch wieder löschen.
- *Senden und Empfangen von Nachrichten:* Benutzer*innen können Textnachrichten an einen Chat-Raum schicken. Diese Nachricht wird in den Chat-Anwendungen von allen Mitgliedern des Raums sofort angezeigt.
- *Senden und Empfangen von privaten Nachrichten:* Benutzer*innen können Nachrichten auch direkt an andere Benutzer*innen senden. Diese müssen dazu nur den Kurznamen des Kommunikationspartners kennen.
- *Darstellung von Chat-Räumen und Nachrichten:* Nachrichten sind chronologisch und übersichtlich darzustellen. Für jede Nachricht ist der Sender, das Sendedatum und der Nachrichtentext anzuzeigen. Nachrichten sind nach Chat-Räumen zu gruppieren.
- *Darstellung von Chat-Verläufen:* Beim Start des Chat-Clients wird für alle Chat-Räume von Benutzer*innen auch ein Teil der Nachrichtenhistorie angezeigt. Wie lange die Historie zurückreicht, kann eingestellt werden.
- *Suchen in Chat-Verläufen:* Benutzer*innen müssen in der Lage sein, im Chat-Verlauf eines Chat-Raums nach Nachrichten, die ein bestimmtes Suchmuster enthalten, zu suchen. In die Suche sind alle Nachrichten eines Chat-Raums einzubeziehen, auch solche, die nicht angezeigt werden.
- *Anzeige von Systembenachrichtigungen:* Benutzer*innen werden vom System über wesentliche Statusänderungen informiert.

Server-Anwendung (ChatBPT-Hub)

- *Nachrichtenvermittlung:* Nachrichten von Chat-Clients müssen empfangen und an die adressierten Chat-Clients weitergeleitet werden.
- *Benutzer*innen-Administration:* Der Server verwaltet die Benutzer*innen-Stammdaten (Name, Kurzname, Passwort) und stellt die notwendigen Lese- und Schreiboperationen zur Verfügung.
- *Authentifizierung:* In der Datenbank sind die für die Authentifizierung von Benutzer*innen erforderlichen Daten zu speichern. Es ist darauf zu achten, dass in der Datenbank die Passwörter nicht im Klartext hinterlegt sind.
- *Administration von Chat-Räumen:* Der Server verwaltet die Chat-Räume, die Zuordnung von Benutzer*innen zu Chat-Räumen sowie den Status der Benutzer*innen in den Chat-Räumen (Erzeuger*in, verbanntes Mitglied).
- *Verlaufsverwaltung:* Nachrichten, die an Chat-Räume geschickt werden, sind in der Datenbank zu speichern. Neben dem Nachrichtentext ist auch das Erstellungsdatum der Nachricht zu persistieren. Die Server-Anwendung muss Möglichkeiten bieten, auf Nachrichten eines Chat-Raums zuzugreifen. Als Filterkriterien können ein Zeitbereich und/oder ein Suchmuster angegeben werden.
- *Senden von Systemnachrichten:* Benutzer*innen werden vom Server über wesentliche Statusänderungen informiert. Dazu zählen: Aufnahme von Benutzer*innen in einen Chat-Raum (Notifikation aller Mitglieder), Verbannung aus einem Chat-Raum (Notifikation aller Mitglieder), Löschen eines Chat-Raums, bei dem Benutzer*innen angemeldet waren, Serveranwendung wurde gestoppt.

Insgesamt bietet ChatBPT eine einfache und flexible Möglichkeit für Benutzer*innen, in Echtzeit zu kommunizieren und zusammenzuarbeiten. Durch die Unterstützung von Chat-Räumen, privaten Nachrichten, Benachrichtigungen, Verlaufsprotokollierung und Benutzer*innen-Management bietet die Anwendung alles, was Benutzer*innen für eine erfolgreiche Online-Kommunikation benötigen.

Ausbaustufe 1: ChatBPT-UI

Der Plattformbetreiber möchte vor der endgültigen Auftragserteilung der Chat-Plattform zunächst einen Prototyp des Chat-Clients sehen.

Entwickeln Sie daher mit Hilfe von JavaFX einen funktionstüchtigen, ausbaufähigen Benutzeroberflächen-Prototyp. Versuchen Sie eine möglichst intuitiv zu verwendende Benutzeroberfläche zu entwerfen. Setzen Sie dafür Ihr in der Usability-LVA erworbenes Wissen ein. Trennen Sie den

Entwurfs- vom Implementierungsprozess, indem Sie Ihre Benutzeroberfläche zunächst mit Mockups (Grobentwurf der Benutzeroberfläche) modellieren. Fügen Sie die Mockups zu Ihrer Systemdokumentation hinzu.

Trennen Sie den Code zur Realisierung der grafischen Benutzeroberfläche vom Code zur Repräsentation der Daten Ihrer Anwendung (Benutzer, Chat-Räume, Nachrichten etc.). Durch diese Maßnahme wird Ihre Anwendung einfach erweiterbar, was Ihnen in der nächsten Ausbaustufe zugutekommen sollte.

Ihr Prototyp soll es ermöglichen, alle Fenster und Dialoge der Benutzeroberfläche zu öffnen (und diese auch wieder zu schließen). Steuerelemente zur Darstellung von Texten sollen bereits mit repräsentativen Testdaten befüllt sein. Die Anwendung muss die Eingaben aber noch nicht über die Programmlaufzeit hinaus speichern können. Als Ersatz dafür können Sie mit hartcodierten Daten arbeiten.

Berücksichtigen Sie bei der Verwaltungsanwendung das Gestaltungsprinzip des responsiven Designs. Das Layout der Anwendung sollte also beim Verändern der Fenstergröße sinnvoll angepasst werden.

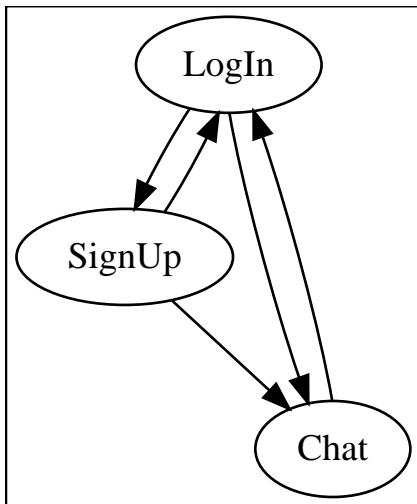
Beachten Sie, dass die Implementierung der anderen Systemkomponenten (Server-Anwendung mit Datenbankbindung) erst in weiteren Ausbaustufen gefordert ist. Die genauen technischen Anforderungen an diese Komponenten werden im weiteren Verlauf der Übung bekannt gegeben.

Lösungsidee/Herangehensweise

Toolwahl:

- Scenebuilder
- Midjourney für Images

Ich dokumentiere das Mockup direkt in Scenebuilder, dazu vorher grundlegen die Idee zum Fluss zwischen den einzelnen Szenen als Graph (wird im Folgenden erweitert, an der Stelle "Chat").



Der Graph stellt die Zusammenhänge der folgenden Controller/FXML-Files dar, die wiederum der ChatBPT-UI-Spezifikation entsprechen.

- LogIn/logIn
- SignUp/signUp
- Chat/chat: fasst erzeugen, beitreten, verbannen (eines Users - Berechtigungskontrolle folgt später dynamisch auf die gleiche Szene), löschen, senden und empfangen (Gruppen- und Privatchat-Funktion), darstellen von Räumen und Nachrichten, darstellen von Verläufen, suchen, anzeigen von Systemnachrichten zusammen, wobei wohl einzelnen Funktionen nur als Button aufrufbar sein werden und die Funktionsweise danach ausgelagert wird.

Chat bezieht Klassen zum Aufbauen sinnvoller Datenstrukturen für die einzelnen Inhalte (User, Nachrichten, Räume) ein, stellt also eine Schnittstelle zwischen UI/Controller und Applikationslogik dar. Ich habe mich letztlich entschieden, folgende Einteilung der Funktions-"räume" zu machen, als einzelnen Szenen.

- Chat-Teilkomponenten
 - Direct-Messaging
 - Group-Chats
 - User-Einstellungen/Sonstiges
 - Anwendungseinstellungen

Im ersten Entwurf in der Oberfläche:



(Über Sign out zurück zur Log In Oberfläche gemäß Zustandsdiagramm.)

Das entspreche m.E. einer sinnvollen Auslagerung der Teilfunktionen gemäß üblicher Chat-Anwendungen.



Die Main-Klasse ChatBPTApplication beinhaltet eine Methode zum Wechseln zwischen Szenen, aber auch eine zum Neuaufstellen der ganzen Stage (die alte wird verworfen): ein Wechsel zwischen Sign Up und Log In ist für mich etwa Szenenwechsel, aber von Log In bzw. Sign Up nach Chat ist Stagewechsel.

Das Designschema folgt folgenden Entscheidungen in Scenebuilder. Details wie Hover-Effekte, etc. durch Ausführung ersichtlich, ebenso sollte der grundlegenden Fluss zwischen den Szenen mit dem Plan zusammenstimmen.

Responsives Design: login und signUp als fixe, einfache und daher nur kleine Fenster implementiert, die Hauptszene Chat als responsiver, komplizierter Anwendungsteil, gemäß der Anforderung von responsive Design.

Implementierung/Design

Log In:





Username

Password

Log In

[Sign Up](#)

Sign Up:



Full Name

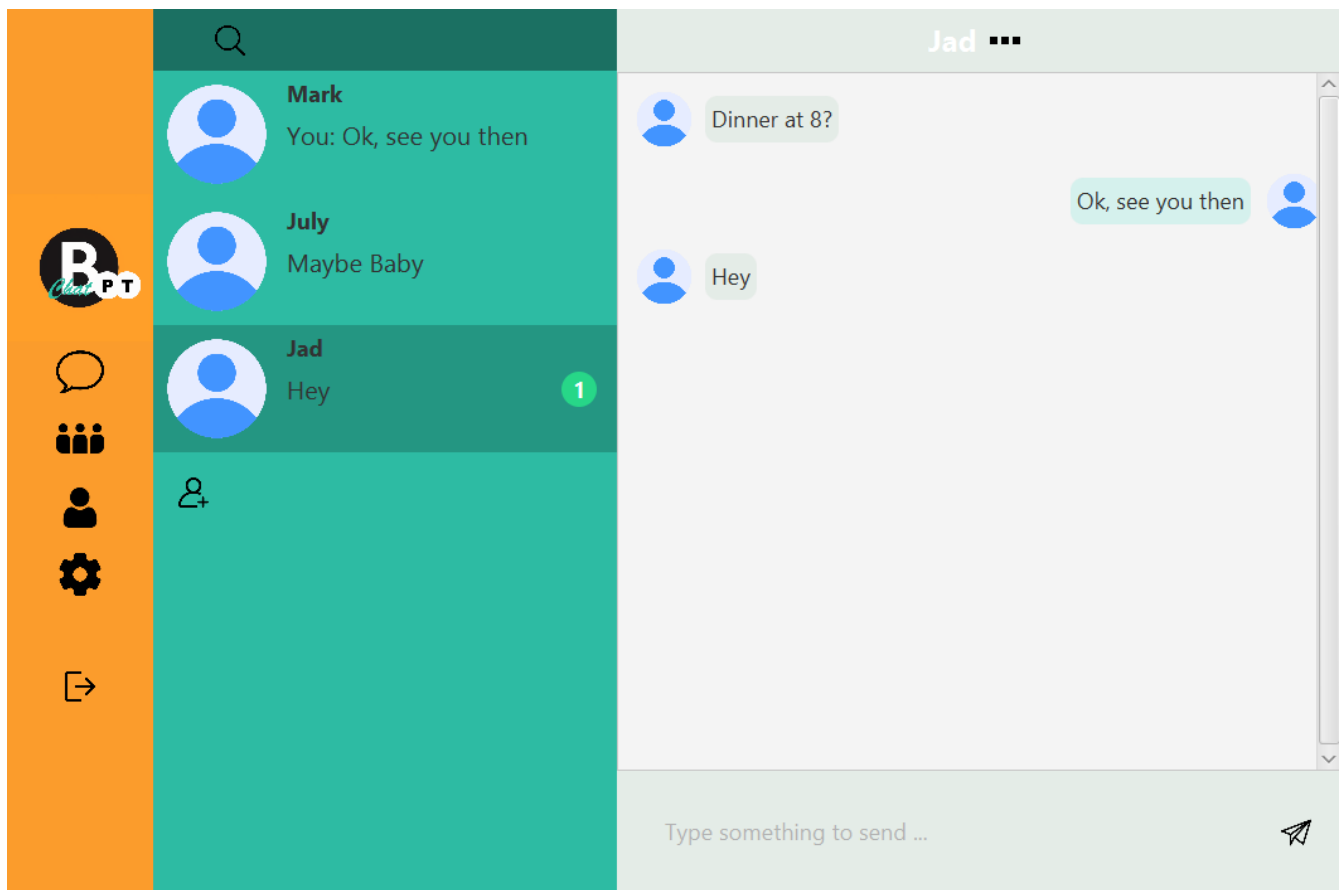
Username

Password

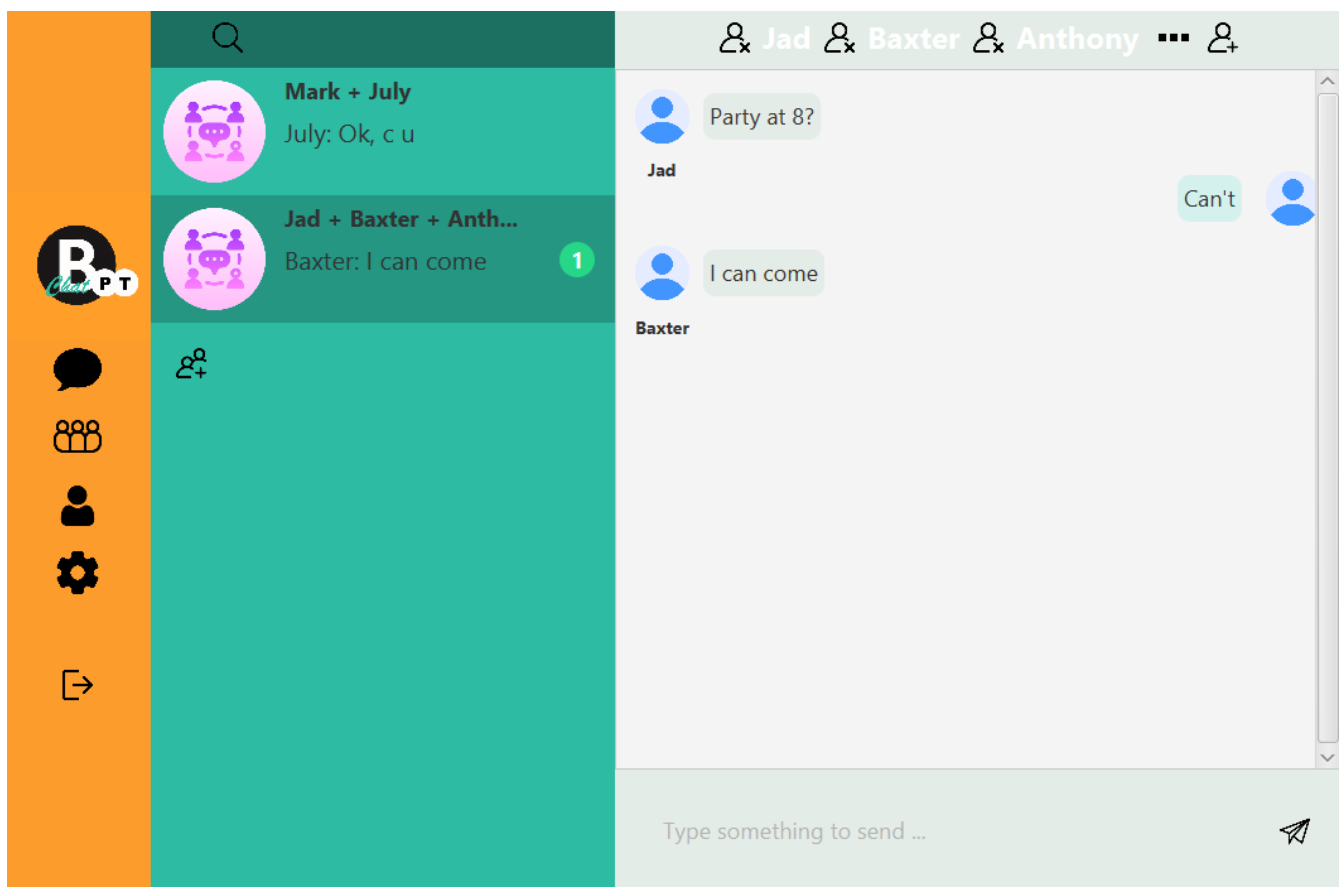
Create User

[Back to Log In](#)

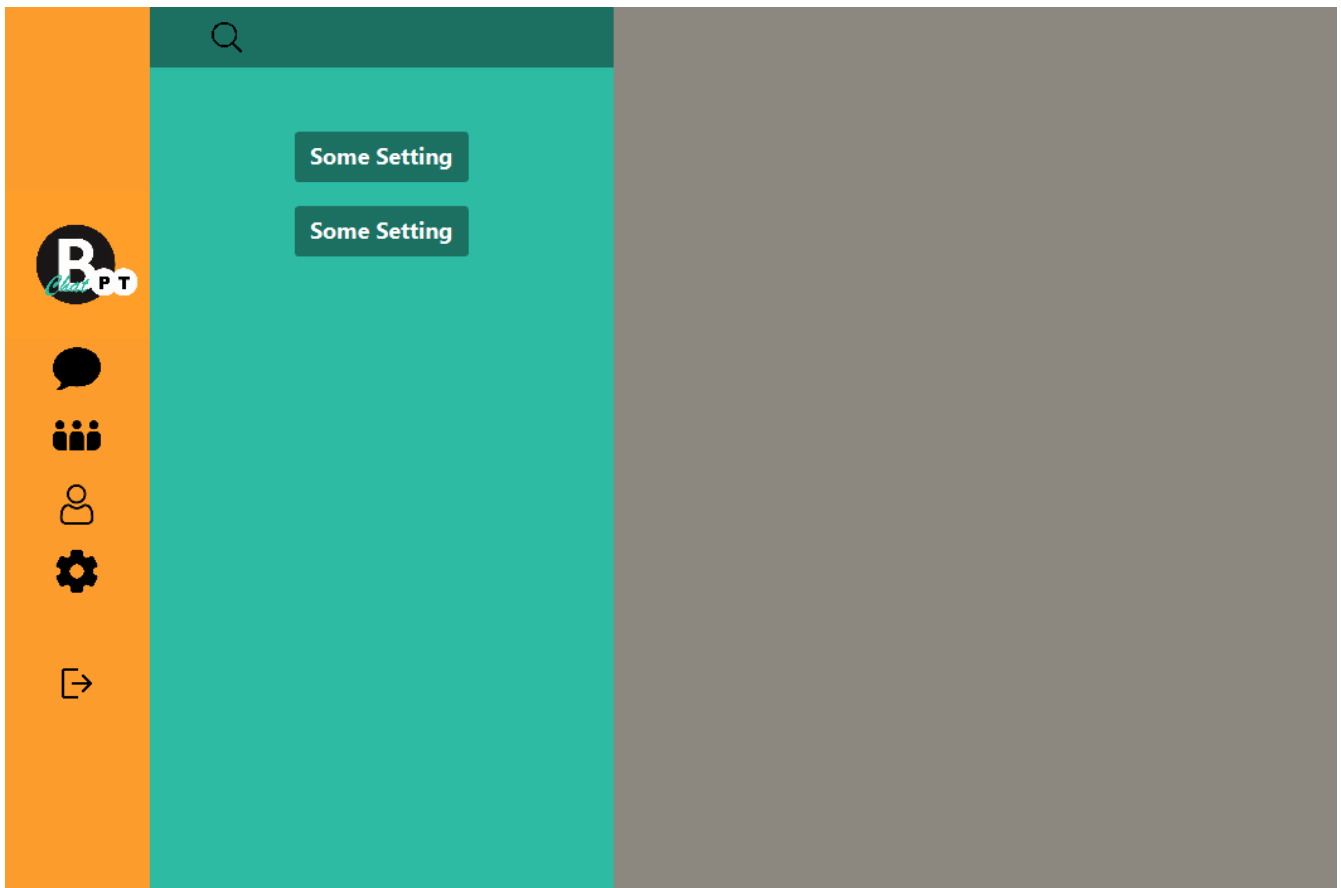
Chat, Direct Messaging (erster Screen):



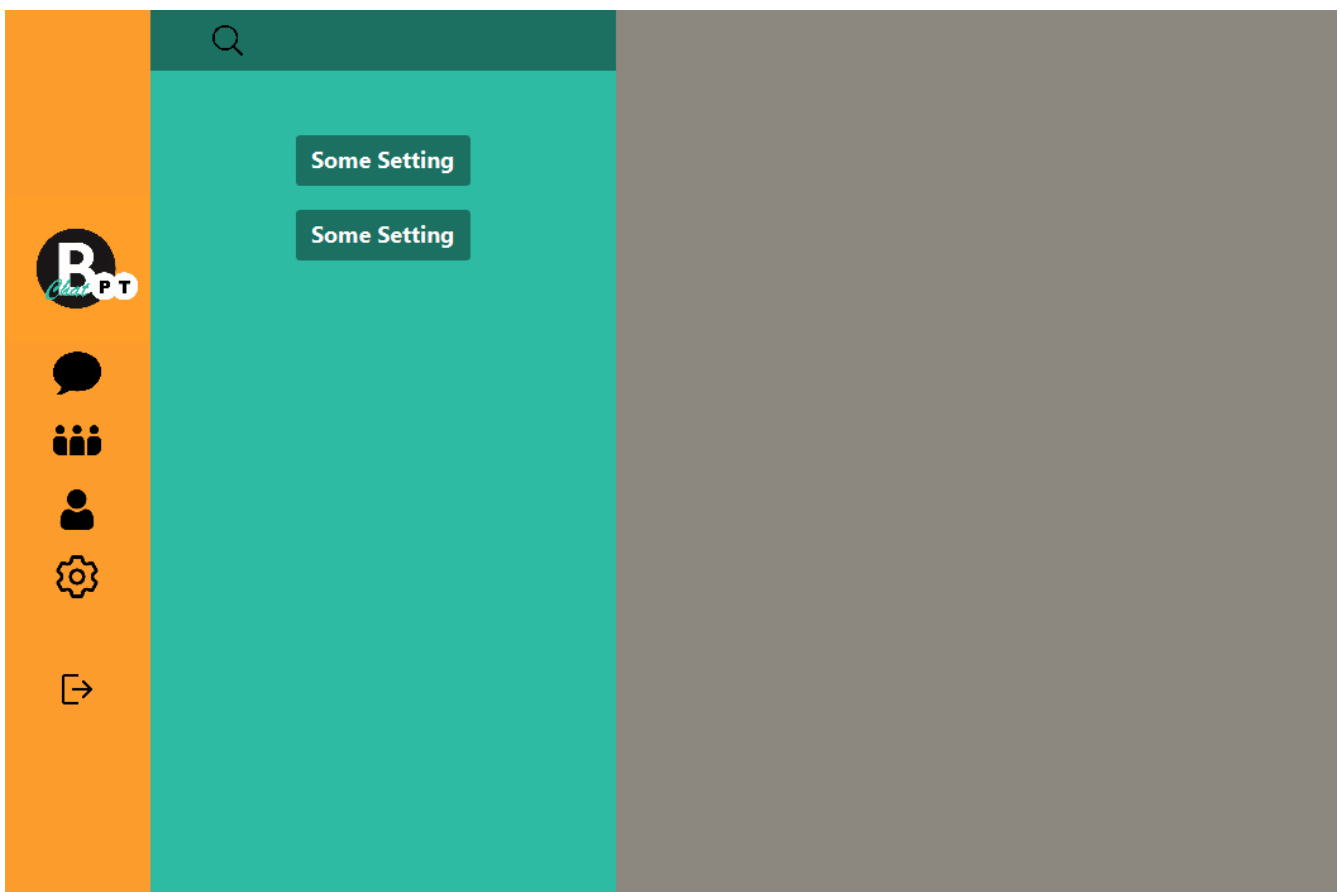
Chat, Group Chats:



Chat, User Settings and Customization:



Chat, App Settings and System Messages:



Anmerkungen zum Design: Um Komplexität zu sparen wurde auf Kontextmenüs weitgehend verzichtet. Das Design erinnert an etwas aus den 70ern oder 80ern: diese App hat nicht den

Anspruch mit WhatsApp o.ä. zu konkurrieren, sonder will einfach sein.

Was den Unterschied zwischen User- und App-Settings betrifft bin ich mir noch nicht sicher, momentan stelle ich mir Profilbild-Einstellung und ähnliches unter User-Settings vor, evtl. in App-Settings die Systemnachrichten.

Was den eingeringelten "1"-er betrifft, um neue Nachrichten zu signalisieren: das ist eigentlich nicht in den Anforderungen und wird vermutlich wieder gestrichen, im Design nur testweise probiert.

Testung

Als ausführbares Program auf Übereinstimmung mit Flussplan getestet. Geringe Abweichungen zwischen SceneBuilder MockUp und Programm hinsichtlich Schriftgröße im Chat und Textpositionierung im Chat gefunden. Das Positionieren von Texten z.B. nach Eingabe einer neuen Chat-Nachricht wird aber noch programmatisch angegriffen, daher glaube ich nicht, dass diese Abweichung problematisch ist.

Folgender Login ist momentan fest-codiert:

- Username: javacoding
- Password: 123

Quellen

SceneBuilder Tutorials

- <https://www.youtube.com/watch?v=VOiFmZyGaps>
- <https://www.youtube.com/watch?v=HBBtlwGpBek>