yourcrush.

Der Amor unter den Liebesrechnern,
wenn du herausfinden möchtest,
wie gut dein Crush zu dir passt.

dokumentation.

ON21
T2 - Frontend
Kathrin Falch

yourcrush.

Inhaltsverzeichnis

app idee.

Love is in the air! Oder vielleicht auch nicht. Wenn wir ehrlich sind, ein wenig Liebe schadet keinem. Und ob man mit seinem Crush zusammenpasst, ist eine Frage, die einem kaum einer beantworten kann, dessen Antwort man aber eigentlich wissen will. Warum dann nicht einfach youcrush. fragen?
Die Grund-Idee der App sollte ein Liebes-Rechner sein, der ausrechnet, zu wieviel Prozent man mit seinem Crush zusammenpasst. Um die Application von anderen ein wenig abzuheben, spielt hierbei das Sternzeichen eine ganz große Rolle. Anstatt einen einfachen Zufallsgenerator laufen zu lassen, bezieht sich der Zufallsgenerator von **yourcrush.** auf die Sternzeichenkonstellationen, die sich in den Feldern ergeben.

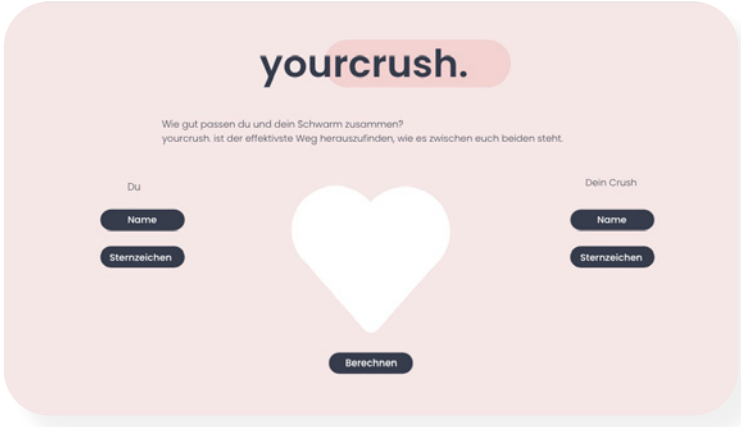
entwicklung.

design.

farben & prototyp

Auf Adobe Xd habe ich Prototypen der Application gebaut. Passend zum Thema bin ich im rosanen Farbschema geblieben und habe als Schriftart ein dunkles blau gewählt. Grundsätzlich sollte das Design einfach und leicht verständlich sein. Außerdem soll der User ein einladendes und freundliches Gefühl beim Nutzen der Application haben.

#F4E3E3 #F0CBCB **#2F3542**



prototyp

pumpendes herz

Um ein wenig Dynamik einzubringen, habe ich im Zentrum der Application ein pumpendes Herz angebracht. Inspiration dafür habe ich mir aus einem Youtube-Tutorial von WebKitCoding [1] geholt.
Durch die CSS-Befehle **box-shadow** und **animation** konnte ich für das Herz einen schattigen Hintergrund erstellen, der das Pumpen des Herzens unterstreicht.
Einzelne Schritte der Animationssequenzen konnte ich durch die CSS-@-Regel **@keyframes** festlegen.

pumpendes Herz

nutzung.

sternzeichenkonstellationen

Beruhend auf einer Studie über die astrologische Partnerwahl [2] und einem Bericht über Sternzeichen, die nicht zueinander passen [3] konnte ich die perfekte Partnerwahl in vier Bereiche unterteilen.
So erhielt ich:

a) Sternzeichen, die sehr gut zueinander passen
b) Sternzeichen, die eher gut zueinander passen
c) Sternzeichen, die eher schlecht zueinander passen
d) Sternzeichen, die sehr schlecht zueinander passen

Die zwölf Sternzeichen konnten ich so in einer Liste genau zuordnen. Für das Einzelprojekt hat der User nun die Möglichkeit, die Namen und die jeweiligen Sternzeichen zu definieren. Außer Acht ließ ich dabei die Abfrage des Geschlechts, das ursprünglich Auswirkungen auf das Sternzeichen gehabt hätte. Jedoch finde ich den Ansatz eher veraltet und ich wollte auch keinem potenziellen Nutzer auf den Schlips treten.

1 https://www.youtube.com/watch?v=xbSkwEbizZk
2 https://www.forschung-und-wissen.de/nachrichten/psychologie/statistik-spricht-fuer-partnerwahl-nach-sternzeichen-13371882
3 https://www.elle.de/horoskop-sternzeichen-passen-nicht-zusammen

Mit rund 140 definierten Konstellationen konnte ich nun also bestimmen, in welchem Bereich sich die Zufallszahlen für die jeweiligen Sternzeichenkonstellationen befinden. Sternzeichen, die sehr gut zueinander passen **(a)**, wies ich einen Erfolgs-Bereich von **75 - 100%** zu. Der Zufallsgenerator wählt in diesem Bereich, wie der Name schon sagt, eine zufällige Zahl. Ebenso gilt das für Sternzeichen, die eher gut zueinander passen **(b)** mit einem Erfolgs-Bereich von **50 - 75%**, sowie **c)** Sternzeichen, die eher schlecht zueinander passen mit Bereich von **25 - 50%**. Zuletzt erhielten Sternzeichen, die sehr schlecht zueinander passen **(d)** einen Erfolgs-Bereich von **0 - 25%**.

75% - 100%		50% - 74%		25% - 49%		0% - 24%	
Steinbock	Fische	Wassermann	Wassermann	Steinbock	Jungfrau	Steinbock	Waage
Fische	Skorpion	Steinbock	Steinbock	Steinbock	Stier	Steinbock	Widder
Zwillinge	Stier	Jungfrau	Jungfrau	Steinbock	Skorpion	Wassermann	Widder
Waage	Widder	Widder	Widder	Wassermann	Waage	Wassermann	Skorpion
Widder	Steinbock	Zwilling	Zwilling	Wassermann	Zwilling	Fische	Löwe
		Schütze	Widder	Fische	Krebs	Fische	Jungfrau
		Stier	Waage	Widder	Wassermann	Fische	Schütze
		Löwe	Widder	Stier	Krebs	Widder	Stier
		Skorpion	Fische	Stier	Jungfrau	Widder	Krebs
				Zwillinge	Waage	Stier	Wassermann
				Zwillinge	Löwe	Zwillinge	Skorpion
				Krebs	Löwe	Krebs	Zwillinge
				Krebs	Skorpion	Krebs	Schütze
				Löwe	Waage	Krebs	Waage
				Jungfrau	Krebs	Löwe	Stier
				Jungfrau	Skorpion	Löwe	Jungfrau
				Schütze	Löwe	Löwe	Skorpion
				Schütze	Waage	Jungfrau	Zwillinge
				Schütze	Wassermann	Jungfrau	Waage
						Jungfrau	Wassermann
						Waage	Skorpion
						Waage	Fische
						Skorpion	Widder
						Skorpion	Stier
						Schütze	Stier
						Schütze	Jungfrau

sternzeichenkonstellationen

Viel Überlegungszeit und Austausch mit Kommilitonen kostete mich hierbei die Frage, wie ich in TypeScript angeben sollte, dass ich ausgewählte Sternzeichen in vier verschiedene Rechnungsbereiche (0 - 25%, 25 - 50%, 50- 75% und 75 -100%) aufteilen kann. Zunächst überlegte ich, jede Konstellation einzeln aufzuzählen, was jedoch sehr viel unnötigen Platz gekostet hätte. Mir war klar, dass sich eine Array-List als am nützlichsten erweisen würde. Schließlich kam ich zu dem Entschluss, vier **Const Assertionen** zu erstellen um jeweils eine **const array** in TypeScript zu deklarieren [1].

1 https://www.codegrepper.com/code-examples/typescript/typescript+const+array+of+objects

addEventListener

Durch die Methode **„addEventListener“** konnte ich viele verschiedene Events antriggern. Beispielsweise verhalf die Methode mir dabei, durch einen Klick auf dem „Jetzt Berechnen“ - Button tatsächlich eine Berechnung auszuführen.
Code-Beispiel: **jetztBerechnen.addEventListener(„click“, berechnen);**

Auch für die Auswahl der Sternzeichen nutzte ich diese Methode.
Code-Beispiel: **elemente[i].addEventListener(„click“, sternzeichenAngeklickt)**

Schließlich sollte der User die Möglichkeit haben, das Ergebnis am Ende auch schließen zu können.
Code-Beispiel: **closeErgebnis.addEventListener(„click“, ergebnisSchliessen);**

if, else if & else

Mit if, else if und else durchläuft die Schleife die Berechnungen A, B, C, D und E. Rechnung E ist dabei die Berechnung, die dann anfällt, sobald der User eine Sternzeichenkombination wählt, die ich nicht definiert habe. Kurz vor Abgabe des Einzelprojektes viel mir noch auf, dass selbst ohne Angabe eines Sternzeichens oder eines Namens eine Zufallszahl durch Rechnung E berechnet und angegeben wird. Diesen kleinen Patzer behob ich dann durch eine Error Meldung, die erscheint, sobald eine Angabe fehlt. Zunächst behob ich also das Problem mit der fehlenden Sternzeichenangabe.
Dazu nutzte ich folgenden Code:

```
let error = false;

if(sternzeichen1 == „Bitte wählen...“ || sternzeichen2 == „Bitte wählen...“) {
    error=true;
    }
```

Sobald also im Eingabefeld „Bitte wählen“ (**sternzeichen-menu**) keine Auswahl getroffen wurde, weist einen das Pop-Up Fenster darauf hin.

Später fiel mir auf, dass auch eine Error-Meldung bei einer fehlenden Eingabe der Namen sinnvoll wäre. Also erweiterte ich die oben genannte If-Schleife um folgenden Code:

```
else if(name1.length == 0 || name2.length == 0){
    error=true;
    }
```

Anhand dieser Erweiterung hatte ich eine simple Lösung: Sollten die eingetragenen Namen (**name1** und **name2**) eine Buchstabenlänge von 0 aufweisen, d.h. eine Eingabe schlichtweg fehlen, so entpuppt sich der Error als true, und die Error-Anzeige ploppt auf.

Mit **else** konnte ich letzten Endes definieren, was im Pop-Up Fenster stehen soll.

```
else{
    alert(„Bitte Namen eingeben und Sternzeichen auswählen.“);
    }
```

Nachdem der User die beiden Namen eingibt und die Sternzeichen auswählt, öffnet sich ein neues Fenster mit dem Ergebnis in Prozentzahl und einem Text, der das Ergebnis kommentiert. Auch dieser wurde mit dem if-Schleifen-Konzept angefertigt. Um das Ergebnis zu visualisierten fügte ich einen Ladebalken hinzu, der sich je nach Prozentzahl füllt.

```
function showErgebnis(punkte: number, name2: string){
    ladebalkenProzent.innerHTML = punkte + ‚%‘;
    ergebnis.classList.add(„in“);
    let widthAsPercent = ladebalkenProzent.clientWidth / 100 * punkte;
    ladebalkenAnteil.style.width = widthAsPercent + ‚px‘;
```

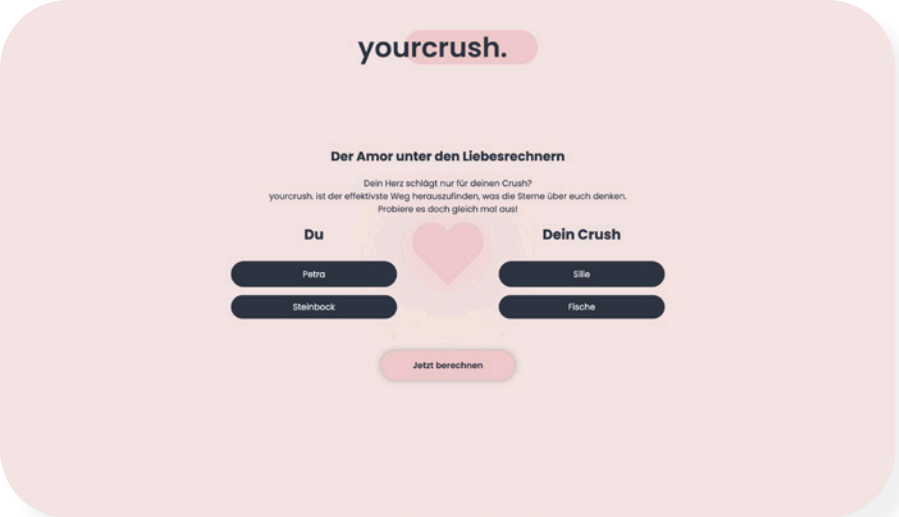
Zum Berechnen des Prozentanteils nutzte ich **Math.floor**. Die Methode gibt die größte Ganzzahl zurück, die größer oder kleiner der angegebenen Berechnung ist.
Bei **Rechnung A** , die eine Zufallszahl zwischen 75 und 100% angeben sollte, nutzte ich beispielsweise folgende Funktion:

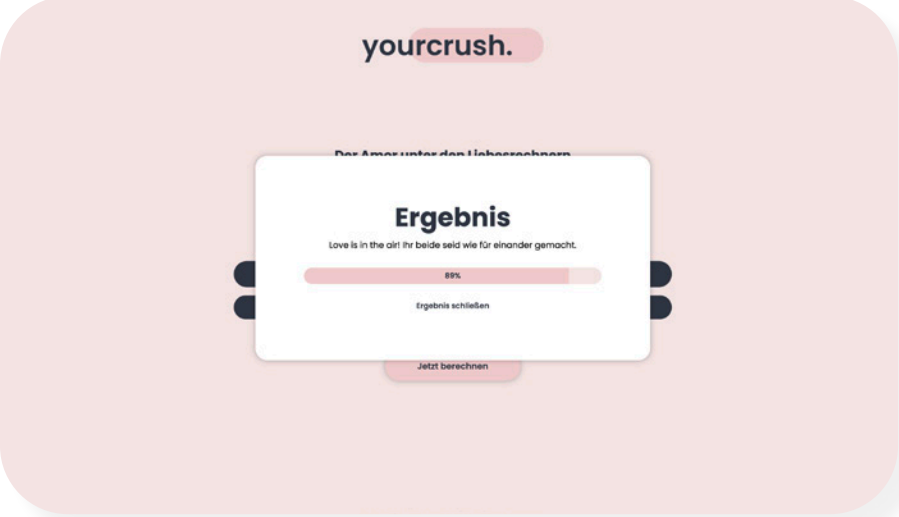
```
function rechnungA(){
    return Math.floor(Math.random() * 26 + 75); }
```

endergebnis.

Wie wendet der User den Liebeskalkulator yourcrush. an?

1. Der User gibt seinen Namen und den Namen seines Crushes in die beiden dafür vorgesehenen Eingabefelder ein.
2. Anschließend hovert er über das darunterliegende elternElement um schließlich die beiden Sternzeichen auszuwählen.
3. Durch einen Klick auf den „Jetzt Berechnen“-Button durchläuft das Programm die Sternzeichenkonstellationen und die dementsprechenden Berechnungen in der If-Schleife.
4. Schließlich erhält der User sein Ergebnis in Prozent, visualisiert durch einen Ladebalken und kann sich zusätzlich einen Kommentar zum Ergebnis durchlesen.





reflexion.

Da die Programmierwelt und vor Allem der komplette Themenbereich rund um **Type-Script** für mich Neuland sind, war für mich von Anfang an klar, dass ich ein Einzelprojekt erstellen möchte, das mir letzten Endes **viel Spaß und Freude beim Erstellen** bereitet. Herauszufinden, in welche Richtung ich demnach gehen möchte, war anfangs sehr schwierig, da ich von Ideen förmlich überflutet wurde und zunächst nicht das passende fand.

Ich suchte nach **verspielten, zeitlosen und lockeren** Themenbereichen die **zwanglos, zügellos und lustig** aufgearbeitet werden können.
Nach ewigem Überlegen viel mir ein Moment mit 11 Jahren ein, bei dem ich vor meinem Laptop saß und auf **Spielaffe.de** den **Love Calculator** ausprobiert habe. Die Eingabe zweier Namen in die Eingabefelder brachten mir schließlich eine Matching-Quote von 34%. Dass es sich hierbei nur um einen **Zufallsgenerator** handelte, war mir zu diesem Zeitpunkt natürlich noch nicht bewusst.

Ganz abgesehen vom Design, das bei vielen Liebesrechnern veraltet ist, war mir das Prinzip des Ausrechnens anhand des simplen Vornamens einfach zu plump.

Also kam ich auf die Idee, den aktuellen Trend der **Sternzeichenkonstellationen** in meinem Einzelprojekt mit aufzunehmen. Dazu recherchierte ich zunächst, welche Sternzeichen **sehr gut, eher gut, eher schlecht und sehr schlecht** zueinander passen. Dabei kamen rund 140 verschiedene Konstellationen heraus.

Zu Beginn fiel mir das Codieren des HTML- und CSS-Codes im Vergleich zum Type-Script-Code eher leicht. Der größte Zeitkiller war und ist das Anwenden von TypeScript in meinem Projekt. Nicht zuletzt, weil ich JavaScript zuvor noch nie angewandt hatte. Jetzt, wo die Application fertig ist, bin ich dankbar, hilfsbereite Kommilitonen zu haben, die sich die Zeit genommen haben, mir bei Fragen zu helfen und schwierige Code-Passagen zu erklären. Besonders das Berechnen der Zufallszahlen war für mich anfangs eine Herausforderung. Im Nachhinein bin ich jedoch froh, die Erfahrung gemacht zu haben, da ich nun einen deutlichen Unterschied zwischen JavaScript und TypeScript erkennen konnte. Tatsächlich tendiere ich jedoch noch eher dazu, Java-Script zu bevorzugen.

quellen.

Zur Informationsbeschaffung und allgemeinen Recherche nutzte ich folgende Quellen:

1. https://www.lovecalculator.com/love.php?name1=Kathrin&name2=Marius
2. https://codepen.io/er-m-k-gupta/pen/ExKNVzg
3. https://www.prokerala.com/entertainment/love-meter/
4. https://www.lovecalculator.club/index-love-score.php
5. https://www.spielaffe.de/Spiel/Love-Tester-2#game_content
6. https://codepen.io/mudrenok/pen/XMobMa
7. https://www.forschung-und-wissen.de/nachrichten/psychologie/statistik-spricht-fuer-partnerwahl-nach-sternzeichen-13371882
8. https://www.elle.de/horoskop-sternzeichen-passen-nicht-zusammen
9. https://www.youtube.com/watch?v=xbSkwEbizZk
10. https://www.codegrepper.com/code-examples/typescript/typescript+const+array+of+objects
11. https://developer.mozilla.org/de/docs/Web/JavaScript/Reference/Global_Objects/Math/random
12. https://lippke.li/typescript-tutorial-deutsch/