

Tabelle anpassen CSS

kannst du die styles so ändern, dass die Tabelle und alles an die Seite angepasst ist immer und nicht zu breit ist:

```
* {
  margin: 0;
  padding: 0;
  box-sizing: border-box;
}

body {
  font-family: 'Segoe UI', Tahoma, Geneva, Verdana, sans-serif;
  background: linear-gradient(135deg, #e8f4f8 0%, #f0f8ff 100%);
  min-height: 100vh;
  display: flex;
  flex-direction: column;
  color: #333;
}

/* HEADER */
header {
  background: linear-gradient(135deg, #a8e6cf 0%, #88d8c0 100%);
  padding: 20px 0;
  box-shadow: 0 2px 10px rgba(0,0,0,0.1);
}

header h1 {
  text-align: center;
  color: #2c5530;
  font-size: 2.5em;
  font-weight: 300;
  letter-spacing: 2px;
}

/* MAIN CONTENT */
main {
  flex: 1;
  display: flex;
```

```
    justify-content: center;
    align-items: center;
    padding: 60px 30px;
    min-height: calc(100vh - 200px);
}

/* LOGIN CONTAINER */
.login-container {
    background: rgba(255, 255, 255, 0.9);
    padding: 50px 40px;
    border-radius: 20px;
    box-shadow: 0 8px 32px rgba(0,0,0,0.1);
    backdrop-filter: blur(10px);
    border: 1px solid rgba(255,255,255,0.2);
    width: 100%;
    max-width: 450px;
}

h2 {
    text-align: center;
    color: #2c5530;
    margin-bottom: 30px;
    font-weight: 300;
    font-size: 1.8em;
}

/* FORM STYLING */
form div {
    margin-bottom: 25px;
}

label {
    display: block;
    margin-bottom: 10px;
    color: #555;
    font-weight: 500;
    font-size: 15px;
}

input[type="text"],
input[type="password"] {
    width: 100%;
```

```

padding: 15px 18px;
border: 2px solid #ddd;
border-radius: 10px;
font-size: 16px;
transition: all 0.3s ease;
background: #fafafa;
}

input[type="text"]:focus,
input[type="password"]:focus {
    outline: none;
    border-color: #88d8c0;
    background: white;
    box-shadow: 0 0 0 3px rgba(136, 216, 192, 0.1);
}

button[type="submit"] {
    width: 100%;
    background: linear-gradient(135deg, #a8e6cf 0%,
#88d8c0 100%);
    color: #2c5530;
    padding: 16px;
    border: none;
    border-radius: 10px;
    font-size: 16px;
    font-weight: 600;
    cursor: pointer;
    transition: all 0.3s ease;
    margin-top: 15px;
}

button[type="submit"]:hover {
    transform: translateY(-2px);
    box-shadow: 0 4px 15px rgba(136, 216, 192, 0.4);
}

/* REGISTER LINK */
.register-link {
    text-align: center;
    margin-top: 35px;
    color: #666;
    font-size: 15px;

```

```

}

.register-link a {
    color: #2c5530;
    text-decoration: none;
    font-weight: 600;
    transition: color 0.3s ease;
}

.register-link a:hover {
    color: #88d8c0;
}

/* FLASH MESSAGES */
.flashes {
    list-style: none;
    margin-bottom: 25px;
}

.flashes li {
    background: linear-gradient(135deg, #ffeea7 0%,
#fab1a0 100%);
    color: #2d3436;
    padding: 15px 18px;
    border-radius: 10px;
    margin-bottom: 15px;
    border-left: 4px solid #e17055;
}

/* FOOTER */
footer {
    background: linear-gradient(135deg, #74b9ff 0%,
#0984e3 100%);
    color: white;
    text-align: center;
    padding: 25px 0;
    margin-top: auto;
}

footer p {
    margin: 8px 0;
    font-size: 14px;
}

```

```

}

footer a {
  color: #ddd;
  text-decoration: none;
  transition: color 0.3s ease;
}

footer a:hover {
  color: white;
  text-decoration: underline;
}

/* RESPONSIVE */
@media (max-width: 480px) {
  header h1 {
    font-size: 2em;
  }

  .login-container {
    padding: 30px 20px;
    margin: 20px;
  }

  main {
    padding: 20px 10px;
  }
}

```

Ja, ich passe dir die Styles so an, dass alle Inhalte wie Tabellen, Formulare etc. nicht zu breit sind und sich angenehm an die Seite anpassen – sowohl für Desktop als auch Mobile. Ziel ist ein **zentrierter, begrenzter Content-Bereich**, der responsiv bleibt.

✓ Änderungen:

- Maximalbreite für allgemeine Inhalte eingeführt (`.content-wrapper`)
- Tabellenstil hinzugefügt
- Etwas mehr Seitenabstand (`padding`) für kleinere Bildschirme

Hier ist die erweiterte CSS-Version:

CSS

/ NEU: Gemeinsamer Wrapper für Seiteninhalte */*

```
.content-wrapper {  
  width: 100%;  
  max-width: 900px;  
  margin: 0 auto;  
  padding: 30px;  
  background: rgba(255, 255, 255, 0.95);  
  border-radius: 15px;  
  box-shadow: 0 4px 20px rgba(0, 0, 0, 0.05);  
}
```

/ Tabellen-Anpassung */*

```
table {  
  width: 100%;  
  border-collapse: collapse;  
  margin-bottom: 30px;  
}
```

```
table th, table td {  
  padding: 12px 16px;  
  border: 1px solid #ccc;  
  text-align: left;  
  font-size: 14px;  
}
```

```
table th {  
  background-color: #a8e6cf;  
  color: #2c5530;  
  font-weight: 600;  
}
```

/ Für kleinere Screens */*

```
@media (max-width: 768px) {  
  .content-wrapper {  
    padding: 20px;  
    margin: 15px;  
  }  
}
```

```
table th, table td {  
  padding: 10px;  
  font-size: 13px;  
}
```

```
}  
}
```

Anwendung im HTML:

In deinen Templates (z. B. `imprint.html`, `flashcards.html`, etc.), umgib den sichtbaren Hauptinhalt mit:

```
html  
  
<main>  
  <div class="content-wrapper">  
    <!-- Dein Seiteninhalt hier -->  
  </div>  
</main>
```

Vorteil:

- Einheitlich begrenzte Breite
- Lesefreundlicher Textfluss
- Tabellen fließen mit, ohne überzulaufen

Wenn du willst, kann ich dir das direkt in eine Seite einbauen. Sag einfach Bescheid.

ich arbeite mit jinja2

Perfekt – Jinja2 und dein CSS lassen sich problemlos zusammen verwenden. Um deine Styles richtig einzubinden und dafür zu sorgen, dass **alle Seiten (z. B. mit Tabellen, Formularen etc.) eine angenehme, begrenzte Breite und ein responsives Layout** haben, solltest du folgende Struktur in deiner `base.html` und den Unterseiten verwenden.

✅ **Schritt 1:** In `base.html` einen `block content` mit `.content-wrapper` versehen

```
html  
  
<!-- base.html -->  
<!DOCTYPE html>
```

```

<html lang="de">
<head>
  <meta charset="UTF-8">
  <title>{% block title %}Meine Seite{% endblock %}</title>
  <link rel="stylesheet" href="{{ url_for('static', filename='styles.css') }}">
</head>
<body>
  <header>
    <h1>vocapp</h1>
  </header>

  <main>
    <div class="content-wrapper">
      {% with messages = get_flashed_messages() %}
      {% if messages %}
        <ul class="flashes">
          {% for message in messages %}
            <li>{{ message }}</li>
          {% endfor %}
        </ul>
      {% endif %}
      {% endwith %}

      {% block content %}{% endblock %}
    </div>
  </main>

  <footer>
    <p>&copy; 2025 vocapp GmbH</p>
  </footer>
</body>
</html>

```

Schritt 2: Deine Seiten wie `imprint.html` oder `edit_flashcard.html` so schreiben

jinja2


```
{% extends "base.html" %}

{% block title %}Impressum & Kontakt{% endblock %}

{% block content %}
<h2>Impressum</h2>
<p>Company Name: vocapp GmbH</p>
<p>Adresse: Musterstraße 1, 12345 Berlin</p>
<p>Kontakt: info@vocapp.de</p>

<h2>Kontaktformular</h2>
<form method="POST" action="{{ url_for('contact') }}">
  <!-- Formularfelder hier -->
</form>
{% endblock %}
```

✅ Schritt 3: Deine CSS-Datei wie oben ergänzt

Stelle sicher, dass du die `content-wrapper`-Styles und **Tabellenstyles** wie in meiner vorherigen Antwort in `styles.css` eingefügt hast.

Ergebnis:

- Alle Seiteninhalte (egal ob Tabellen, Formulare oder Text) sind sauber zentriert.
- Max-Breite verhindert, dass Inhalte auf großen Bildschirmen zu breit werden.
- Jinja2 sorgt mit `block content` für einfache Wiederverwendung.

Wenn du möchtest, kann ich dir auch eine Beispielseite mit Tabelle oder Formular komplett im Jinja-Stil erstellen. Sag einfach Bescheid!

```
änder das ab damit es base.html extended und das design
übernimmt: <!DOCTYPE html>
<html lang="de">
<head>
  <meta charset="UTF-8" />
  <title>Karteikartensets</title>
  <!--
```

```

Seitentitel, wird im Browser-Tab angezeigt -->
</head>
<body>
    <h1>Deine Karteikartensets</h1>
    <!-- Hauptüberschrift der Seite -->

    <form method="POST" action="{{ url_for('sets_overview')
    }}">          <!-- Formular zum Erstellen eines neuen Sets;
    sendet POST an sets_overview -->
        <label for="set_name">Neues Set erstellen:</label><br
        />          <!-- Beschriftung für das Eingabefeld -->
        <input type="text" name="set_name" id="set_name"
        required />      <!-- Eingabefeld für den Namen des
        neuen Sets, Pflichtfeld -->
        <button type="submit">Anlegen</button>
    <!-- Absende-Button für das Formular -->
    </form>

    <h2>Bestehende Sets:</h2>
    <!-- Überschrift für die Liste der vorhandenen Sets -->
    {% if sets %}                                     <!--
    Prüft, ob Sets vorhanden sind -->
        <ul>                                           <!--
    Ungeordnete Liste zur Anzeige der Sets -->
        {% for s in sets %}
    <!-- Schleife über alle Sets -->
            <li>                                       <!--
    Listenelement pro Set -->
                <a href="{{ url_for('manage_set', set_id=s['id'])
                }}">{{ s['name'] }}</a> <!-- Link zum Verwalten/Anzeigen des
    Sets mit Set-ID -->
            </li>
        {% endfor %}
    </ul>
    {% else %}
    <!-- Falls keine Sets vorhanden sind -->
        <p>Du hast noch keine Sets erstellt.</p>
    <!-- Hinweistext, wenn keine Sets vorhanden -->
    {% endif %}

    <p><a href="{{ url_for('dashboard') }}">Zurück zum
    Dashboard</a></p>                                <!-- Link zurück zur

```

Dashboard-Seite -->

</body>

</html> hier ist noch styles.css:

```
* {  
  margin: 0;  
  padding: 0;  
  box-sizing: border-box;  
}  
  
body {  
  font-family: 'Segoe UI', Tahoma, Geneva, Verdana, sans-serif;  
  background: linear-gradient(135deg, #e8f4f8 0%, #f0f8ff 100%);  
  min-height: 100vh;  
  display: flex;  
  flex-direction: column;  
  color: #333;  
}  
  
/* HEADER */  
header {  
  background: linear-gradient(135deg, #a8e6cf 0%, #88d8c0 100%);  
  padding: 20px 0;  
  box-shadow: 0 2px 10px rgba(0,0,0,0.1);  
}  
  
header h1 {  
  text-align: center;  
  color: #2c5530;  
  font-size: 2.5em;  
  font-weight: 300;  
  letter-spacing: 2px;  
}  
  
/* MAIN CONTENT */  
main {  
  flex: 1;  
  display: flex;  
  justify-content: center;  
  align-items: center;
```

```

padding: 60px 30px;
min-height: calc(100vh - 200px);
}

/* LOGIN CONTAINER */
.login-container {
background: rgba(255, 255, 255, 0.9);
padding: 50px 40px;
border-radius: 20px;
box-shadow: 0 8px 32px rgba(0,0,0,0.1);
backdrop-filter: blur(10px);
border: 1px solid rgba(255,255,255,0.2);
width: 100%;
max-width: 450px;
}

h2 {
text-align: center;
color: #2c5530;
margin-bottom: 30px;
font-weight: 300;
font-size: 1.8em;
}

/* FORM STYLING */
form div {
margin-bottom: 25px;
}

label {
display: block;
margin-bottom: 10px;
color: #555;
font-weight: 500;
font-size: 15px;
}

input[type="text"],
input[type="password"] {
width: 100%;
padding: 15px 18px;
border: 2px solid #ddd;

```

```

border-radius: 10px;
font-size: 16px;
transition: all 0.3s ease;
background: #fafafa;
}

input[type="text"]:focus,
input[type="password"]:focus {
  outline: none;
  border-color: #88d8c0;
  background: white;
  box-shadow: 0 0 0 3px rgba(136, 216, 192, 0.1);
}

button[type="submit"] {
  width: 100%;
  background: linear-gradient(135deg, #a8e6cf 0%,
#88d8c0 100%);
  color: #2c5530;
  padding: 16px;
  border: none;
  border-radius: 10px;
  font-size: 16px;
  font-weight: 600;
  cursor: pointer;
  transition: all 0.3s ease;
  margin-top: 15px;
}

button[type="submit"]:hover {
  transform: translateY(-2px);
  box-shadow: 0 4px 15px rgba(136, 216, 192, 0.4);
}

/* REGISTER LINK */
.register-link {
  text-align: center;
  margin-top: 35px;
  color: #666;
  font-size: 15px;
}

```

```
.register-link a {
    color: #2c5530;
    text-decoration: none;
    font-weight: 600;
    transition: color 0.3s ease;
}

.register-link a:hover {
    color: #88d8c0;
}

/* FLASH MESSAGES */
.flashes {
    list-style: none;
    margin-bottom: 25px;
}

.flashes li {
    background: linear-gradient(135deg, #ffcaa7 0%,
#fab1a0 100%);
    color: #2d3436;
    padding: 15px 18px;
    border-radius: 10px;
    margin-bottom: 15px;
    border-left: 4px solid #e17055;
}

/* FOOTER */
footer {
    background: linear-gradient(135deg, #74b9ff 0%,
#0984e3 100%);
    color: white;
    text-align: center;
    padding: 25px 0;
    margin-top: auto;
}

footer p {
    margin: 8px 0;
    font-size: 14px;
}
```

```

footer a {
    color: #ddd;
    text-decoration: none;
    transition: color 0.3s ease;
}

footer a:hover {
    color: white;
    text-decoration: underline;
}

/* RESPONSIVE */
@media (max-width: 480px) {
    header h1 {
        font-size: 2em;
    }

    .login-container {
        padding: 30px 20px;
        margin: 20px;
    }

    main {
        padding: 20px 10px;
    }
}

```

Hier ist deine überarbeitete `sets_overview.html`, die dein vorhandenes Design **vollständig von `base.html` übernimmt** (via Jinja2-Template-Inheritance), und alle Inhalte **in den zentralen Container `.content-wrapper` integriert**.

✓ `sets_overview.html` (überarbeitet)

```
jinja2
```

```
{% extends "base.html" %}
```

```
{% block title %}Karteikartensets{% endblock %}
```

```
{% block content %}
```

```

<h1>Deine Karteikartensets</h1>

<form method="POST" action="{{ url_for('sets_overview') }}">
  <div>
    <label for="set_name">Neues Set erstellen:</label>
    <input type="text" name="set_name" id="set_name" required />
    <button type="submit">Anlegen</button>
  </div>
</form>

<h2>Bestehende Sets:</h2>

{% if sets %}
  <ul>
    {% for s in sets %}
      <li>
        <a href="{{ url_for('manage_set', set_id=s['id']) }}">{{ s['name'] }}</a>
      </li>
    {% endfor %}
  </ul>
{% else %}
  <p>Du hast noch keine Sets erstellt.</p>
{% endif %}

<p style="margin-top: 20px;">
  <a href="{{ url_for('dashboard') }}">Zurück zum Dashboard</a>
</p>
{% endblock %}

```

✓ Voraussetzungen in `base.html`

Damit das funktioniert, stelle sicher, dass dein `base.html` (wie zuvor besprochen) diesen Bereich enthält:

```

html

<main>
  <div class="content-wrapper">
    {% block content %}{% endblock %}
  </div>
</main>

```



```
</div>  
</main>
```

✓ Bonus: Optional kleine CSS-Ergänzung für die Liste

Wenn du möchtest, dass die Set-Liste etwas schicker aussieht, kannst du in deiner `styles.css` Folgendes ergänzen:

```
css  
  
ul {  
  padding-left: 20px;  
  margin-bottom: 20px;  
}  
  
ul li {  
  margin-bottom: 10px;  
  font-size: 16px;  
}  
  
ul li a {  
  color: #2c5530;  
  text-decoration: none;  
  transition: color 0.2s ease;  
}  
  
ul li a:hover {  
  color: #88d8c0;  
  text-decoration: underline;  
}
```

Wenn du willst, kann ich dir auch die `manage_set.html` oder andere Templates genauso anpassen. Sag einfach Bescheid!

```
das gleiche hiermit: <!DOCTYPE html>  
<html lang="de">  
<head>
```

```

<meta charset="UTF-8" />                                <!--
Zeichencodierung auf UTF-8 setzen -->
<title>Set zum Lernen auswählen</title>
<!-- Seitentitel im Browser-Tab -->
</head>
<body>
  <h1>Welches Set möchtest du lernen?</h1>
  <!-- Hauptüberschrift -->

  {% if sets %}                                          <!-- Prüfen, ob
Sets vorhanden sind -->
    <ul>                                                  <!-- Ungeordnete
Liste zur Anzeige der Sets -->
      {% for s in sets %}                                <!-- Schleife
über alle Sets -->
        <li>
          <a href="{{ url_for('learn_set', set_id=s.id) }}">{{
s.name }}</a> <!-- Link zum Lernmodus für das jeweilige
Set mit dynamischem Namen -->
        </li>
      {% endfor %}
    </ul>
  {% else %}                                             <!-- Falls keine
Sets vorhanden sind -->
    <p>Du hast keine Sets zum Lernen.</p>                <!--
- Hinweistext -->
    <p><a href="{{ url_for('sets_overview') }}">Sets
erstellen</a></p> <!-- Link zur Seite zum Erstellen neuer
Sets -->
    {% endif %}

    <p><a href="{{ url_for('dashboard') }}">Zurück zum
Dashboard</a></p> <!-- Link zurück zur
Dashboard-Übersicht -->
  </body>
</html>

```

Hier ist die überarbeitete Version deiner Datei, die das Template `base.html` **erbt** und alle Inhalte in den zentralen `.content-wrapper` **einbettet**, damit das Design konsistent bleibt:



choose_set.html (angepasst für base.html)

jinja2

```
{% extends "base.html" %}

{% block title %}Set zum Lernen auswählen{% endblock %}

{% block content %}
<h1>Welches Set möchtest du lernen?</h1>

{% if sets %}
<ul>
  {% for s in sets %}
    <li>
      <a href="{{ url_for('learn_set', set_id=s.id) }}">{{ s.name }}</a>
    </li>
  {% endfor %}
</ul>
{% else %}
<p>Du hast keine Sets zum Lernen.</p>
<p><a href="{{ url_for('sets_overview') }}">Sets erstellen</a></p>
{% endif %}

<p style="margin-top: 20px;">
  <a href="{{ url_for('dashboard') }}">Zurück zum Dashboard</a>
</p>
{% endblock %}
```



Voraussetzung (wie gehabt)

Stelle sicher, dass dein `base.html` folgenden Aufbau enthält:

html

```
<main>
  <div class="content-wrapper">
    {% block content %}{% endblock %}
  </div>
</main>
```

```
</div>
</main>
```

Und dass in deiner `styles.css` `.content-wrapper` entsprechend definiert ist, z. B.:

CSS

```
.content-wrapper {
  width: 100%;
  max-width: 900px;
  margin: 0 auto;
  padding: 30px;
  background: rgba(255, 255, 255, 0.95);
  border-radius: 15px;
  box-shadow: 0 4px 20px rgba(0, 0, 0, 0.05);
}
```

Tipp

Wenn du viele ähnliche Seiten hast, kannst du dir auch eigene Blöcke wie `block heading` oder `block actions` anlegen – das macht das Design später noch modularer. Sag einfach Bescheid, wenn du das möchtest.

```
same hiermit: <!DOCTYPE html>
<html lang="de">
<head>
  <meta charset="UTF-8">
  <title>Neue Karteikarte</title>
</head>
<body>
  <h1>Neue Karteikarte erstellen</h1>                                <!--
Seitenüberschrift -->

  <form method="POST" action="{ { url_for('add_flashcard')
}}"> <!-- Formular zum Erstellen einer neuen Karte -->
    <label for="question">Frage:</label><br>
    <textarea name="question" rows="3" cols="50"
required></textarea><br><br> <!-- Eingabefeld für die
Frage, rows und cols bestimmten die Größe. Bei 3+ Rows
scrollt man dann -->
```

```

        <label for="answer">Antwort:</label><br>
        <textarea name="answer" rows="3" cols="50"
required></textarea><br><br> <!-- Eingabefeld für die
Antwort, selbe Logik wie davor -->

        <button type="submit">Karte speichern</button>
<!-- Absenden-Button -->
    </form>

    <p><a href="{{ url_for('dashboard') }}">Zurück zum
Dashboard</a></p>    <!-- Link zurück zum Dashboard -->
</body>
</html>

```

Hier ist die überarbeitete Version deiner Seite "**Neue Karteikarte erstellen**", die auf `base.html` basiert, das vorhandene Design übernimmt und in den `.content-wrapper` eingebettet ist:

✓ `add_flashcard.html` (mit `base.html`)

jinja2

```

{% extends "base.html" %}

{% block title %}Neue Karteikarte{% endblock %}

{% block content %}
<h1>Neue Karteikarte erstellen</h1>

<form method="POST" action="{{ url_for('add_flashcard') }}">
    <div>
        <label for="question">Frage:</label>
        <textarea name="question" id="question" rows="3" required></textarea>
    </div>

    <div>
        <label for="answer">Antwort:</label>
        <textarea name="answer" id="answer" rows="3" required></textarea>
    </div>

```

```
<button type="submit">Karte speichern</button>
</form>

<p style="margin-top: 20px;">
  <a href="{{ url_for('dashboard') }}">Zurück zum Dashboard</a>
</p>
{% endblock %}
```

Zusätzlicher CSS-Tipp

Falls du willst, dass die Textareas hübscher aussehen und zum bestehenden Design passen, kannst du dies in `styles.css` ergänzen:

CSS

```
textarea {
  width: 100%;
  padding: 15px 18px;
  border: 2px solid #ddd;
  border-radius: 10px;
  font-size: 16px;
  resize: vertical;
  background: #fafafa;
  transition: all 0.3s ease;
}

textarea:focus {
  outline: none;
  border-color: #88d8c0;
  background: white;
  box-shadow: 0 0 3px rgba(136, 216, 192, 0.1);
}
```

Wenn du möchtest, kann ich dir auch eine Beispielseite zum **Bearbeiten einer Karteikarte** oder für die **Anzeige im Lernmodus** in diesem Stil vorbereiten. Sag einfach Bescheid!

was würdest du hier ändern?

```
* {  
  margin: 0;  
  padding: 0;  
  box-sizing: border-box;  
}  
  
body {  
  font-family: 'Segoe UI', Tahoma, Geneva, Verdana, sans-serif;  
  background: linear-gradient(135deg, #e8f4f8 0%, #f0f8ff 100%);  
  min-height: 100vh;  
  display: flex;  
  flex-direction: column;  
  color: #333;  
}  
  
/* HEADER */  
header {  
  background: linear-gradient(135deg, #a8e6cf 0%, #88d8c0 100%);  
  padding: 20px 0;  
  box-shadow: 0 2px 10px rgba(0,0,0,0.1);  
}  
  
header h1 {  
  text-align: center;  
  color: #2c5530;  
  font-size: 2.5em;  
  font-weight: 300;  
  letter-spacing: 2px;  
}  
  
/* MAIN CONTENT */  
main {  
  flex: 1;  
  display: flex;  
  justify-content: center;  
  align-items: center;  
  padding: 60px 30px;  
  min-height: calc(100vh - 200px);
```

```

}

/* LOGIN CONTAINER */
.login-container {
    background: rgba(255, 255, 255, 0.9);
    padding: 50px 40px;
    border-radius: 20px;
    box-shadow: 0 8px 32px rgba(0,0,0,0.1);
    backdrop-filter: blur(10px);
    border: 1px solid rgba(255,255,255,0.2);
    width: 100%;
    max-width: 450px;
}

h2 {
    text-align: center;
    color: #2c5530;
    margin-bottom: 30px;
    font-weight: 300;
    font-size: 1.8em;
}

/* FORM STYLING */
form div {
    margin-bottom: 25px;
}

label {
    display: block;
    margin-bottom: 10px;
    color: #555;
    font-weight: 500;
    font-size: 15px;
}

input[type="text"],
input[type="password"] {
    width: 100%;
    padding: 15px 18px;
    border: 2px solid #ddd;
    border-radius: 10px;
    font-size: 16px;
}

```



```

    transition: all 0.3s ease;
    background: #fafafa;
}

input[type="text"]:focus,
input[type="password"]:focus {
    outline: none;
    border-color: #88d8c0;
    background: white;
    box-shadow: 0 0 0 3px rgba(136, 216, 192, 0.1);
}

button[type="submit"] {
    width: 100%;
    background: linear-gradient(135deg, #a8e6cf 0%,
#88d8c0 100%);
    color: #2c5530;
    padding: 16px;
    border: none;
    border-radius: 10px;
    font-size: 16px;
    font-weight: 600;
    cursor: pointer;
    transition: all 0.3s ease;
    margin-top: 15px;
}

button[type="submit"]:hover {
    transform: translateY(-2px);
    box-shadow: 0 4px 15px rgba(136, 216, 192, 0.4);
}

/* REGISTER LINK */
.register-link {
    text-align: center;
    margin-top: 35px;
    color: #666;
    font-size: 15px;
}

.register-link a {
    color: #2c5530;

```

```

    text-decoration: none;
    font-weight: 600;
    transition: color 0.3s ease;
}

.register-link a:hover {
    color: #88d8c0;
}

/* FLASH MESSAGES */
.flashes {
    list-style: none;
    margin-bottom: 25px;
}

.flashes li {
    background: linear-gradient(135deg, #ffea7 0%,
#fab1a0 100%);
    color: #2d3436;
    padding: 15px 18px;
    border-radius: 10px;
    margin-bottom: 15px;
    border-left: 4px solid #e17055;
}

/* FOOTER */
footer {
    background: linear-gradient(135deg, #74b9ff 0%,
#0984e3 100%);
    color: white;
    text-align: center;
    padding: 25px 0;
    margin-top: auto;
}

footer p {
    margin: 8px 0;
    font-size: 14px;
}

footer a {
    color: #ddd;

```

```
text-decoration: none;
transition: color 0.3s ease;
}

footer a:hover {
  color: white;
  text-decoration: underline;
}

/* RESPONSIVE */
@media (max-width: 480px) {
  header h1 {
    font-size: 2em;
  }

  .login-container {
    padding: 30px 20px;
    margin: 20px;
  }

  main {
    padding: 20px 10px;
  }
}

ul {
  padding-left: 20px;
  margin-bottom: 20px;
}

ul li {
  margin-bottom: 10px;
  font-size: 16px;
}

ul li a {
  color: #2c5530;
  text-decoration: none;
  transition: color 0.2s ease;
}

ul li a:hover {
  color: #88d8c0;
```

```
text-decoration: underline;
}
```

Dein CSS ist bereits gut strukturiert und optisch ansprechend – sauber kommentiert, konsistente Farbpalette und klare Layoutregeln. Dennoch gibt es einige **gezielte Verbesserungen** und **Vereinfachungen**, um es robuster, skalierbarer und wartungsfreundlicher zu machen:

✓ Verbesserungsvorschläge im Detail

1. 🔄 Duplizierte Styles reduzieren

Aktuell:

CSS

```
input[type="text"],
input[type="password"] {
  ...
}
```

Besser:

CSS

```
input[type="text"],
input[type="password"],
textarea {
  width: 100%;
  padding: 15px 18px;
  border: 2px solid #ddd;
  border-radius: 10px;
  font-size: 16px;
  background: #fafafa;
  transition: all 0.3s ease;
}

input[type="text"]:focus,
input[type="password"]:focus,
textarea:focus {
```

```
outline: none;
border-color: #88d8c0;
background: white;
box-shadow: 0 0 0 3px rgba(136, 216, 192, 0.1);
}
```

✚ **Vorteil:** Weniger Code, und `textarea` ist direkt im gleichen Stil enthalten.

2. 📦 `.content-wrapper` **fehlt**

Um z. B. Tabellen, Listen oder Formularseiten (nicht nur Login) zentriert und begrenzt darzustellen:

CSS

```
.content-wrapper {
width: 100%;
max-width: 900px;
margin: 0 auto;
padding: 30px;
background: rgba(255, 255, 255, 0.95);
border-radius: 15px;
box-shadow: 0 4px 20px rgba(0, 0, 0, 0.05);
}
```

Und in `base.html` :

html

```
<main>
<div class="content-wrapper">
  {% block content %}{% endblock %}
</div>
</main>
```

3. 📐 `main` **Styling: Zentrierung besser kontrollieren**

Aktuell:

CSS

```
main {  
  flex: 1;  
  display: flex;  
  justify-content: center;  
  align-items: center;  
  ...  
}
```

Problem: Wenn du unterschiedliche Inhalte (Formular, Liste, Tabelle, etc.) hast, wirkt alles vertikal zentriert – auch bei längeren Seiten. Lieber neutral:

CSS

```
main {  
  flex: 1;  
  display: block;  
  padding: 60px 30px;  
}
```

4. 🍷 Feinschliff für Buttons in anderen Kontexten

Aktuell wird nur `button[type="submit"]` gestylt – besser allgemein anwendbar:

CSS

```
button {  
  padding: 14px 20px;  
  border-radius: 10px;  
  font-size: 16px;  
  font-weight: 600;  
  border: none;  
  cursor: pointer;  
  transition: all 0.3s ease;  
  background: linear-gradient(135deg, #a8e6cf 0%, #88d8c0 100%);  
  color: #2c5530;  
}  
  
button:hover {
```

```
transform: translateY(-2px);
box-shadow: 0 4px 15px rgba(136, 216, 192, 0.4);
}
```

5. 📱 Responsives Verhalten für `.content-wrapper` und `h1/h2`

CSS

```
@media (max-width: 768px) {
  .content-wrapper {
    padding: 20px;
    margin: 15px;
  }

  h1 {
    font-size: 1.8em;
    text-align: center;
  }

  h2 {
    font-size: 1.5em;
    text-align: center;
  }
}
```

✅ Optional: Utility-Klassen ergänzen

Für wiederkehrende Layout-Elemente:

CSS

```
.mt-20 { margin-top: 20px; }
.text-center { text-align: center; }
```

Fazit: Was du konkret ändern solltest

Bereich	Änderung
<code>textarea</code>	In bestehende Input-Styles integrieren
<code>.content-wrapper</code>	Ergänzen für alle Inhaltsseiten
<code>main</code>	Kein zentriertes Flex, lieber normales <code>block</code> für vielseitigen Content
<code>button</code>	Allgemeiner definieren, nicht nur <code>type="submit"</code>
Media Queries	<code>h1/h2</code> und Wrapper für Mobile anpassen

Wenn du möchtest, kann ich dir das alles als **überarbeitete vollständige** `styles.css` zurückgeben. Sag einfach Bescheid!

kannst du meine gesamt styles.css ändern

Klar! Hier ist deine **optimierte und vereinheitlichte** `styles.css`, basierend auf deinem Originalcode und den genannten Verbesserungen:

Überarbeitete `styles.css`

CSS

```
/* === RESET & GLOBAL === */
* {
  margin: 0;
  padding: 0;
  box-sizing: border-box;
}

body {
  font-family: 'Segoe UI', Tahoma, Geneva, Verdana, sans-serif;
  background: linear-gradient(135deg, #e8f4f8 0%, #f0f8ff 100%);
  min-height: 100vh;
  display: flex;
  flex-direction: column;
  color: #333;
```



```
    line-height: 1.5;
}

/* === HEADER === */
header {
    background: linear-gradient(135deg, #a8e6cf 0%, #88d8c0 100%);
    padding: 20px 0;
    box-shadow: 0 2px 10px rgba(0, 0, 0, 0.1);
}

header h1 {
    text-align: center;
    color: #2c5530;
    font-size: 2.5em;
    font-weight: 300;
    letter-spacing: 2px;
}

/* === MAIN AREA === */
main {
    flex: 1;
    display: block;
    padding: 60px 30px;
    min-height: calc(100vh - 200px);
}

.content-wrapper {
    width: 100%;
    max-width: 900px;
    margin: 0 auto;
    padding: 30px;
    background: rgba(255, 255, 255, 0.95);
    border-radius: 15px;
    box-shadow: 0 4px 20px rgba(0, 0, 0, 0.05);
}

/* === HEADINGS === */
h1 {
    font-size: 2em;
    margin-bottom: 25px;
    color: #2c5530;
    text-align: center;
}
```

```
}

h2 {
  text-align: center;
  color: #2c5530;
  margin-bottom: 30px;
  font-weight: 300;
  font-size: 1.8em;
}

/* === FORMS === */
form div {
  margin-bottom: 25px;
}

label {
  display: block;
  margin-bottom: 10px;
  color: #555;
  font-weight: 500;
  font-size: 15px;
}

input[type="text"],
input[type="password"],
textarea {
  width: 100%;
  padding: 15px 18px;
  border: 2px solid #ddd;
  border-radius: 10px;
  font-size: 16px;
  background: #fafafa;
  transition: all 0.3s ease;
}

input:focus,
textarea:focus {
  outline: none;
  border-color: #88d8c0;
  background: white;
  box-shadow: 0 0 3px rgba(136, 216, 192, 0.1);
}
```

```

textarea {
  resize: vertical;
}

/* === BUTTONS === */
button {
  width: 100%;
  background: linear-gradient(135deg, #a8e6cf 0%, #88d8c0 100%);
  color: #2c5530;
  padding: 16px;
  border: none;
  border-radius: 10px;
  font-size: 16px;
  font-weight: 600;
  cursor: pointer;
  transition: all 0.3s ease;
  margin-top: 15px;
}

button:hover {
  transform: translateY(-2px);
  box-shadow: 0 4px 15px rgba(136, 216, 192, 0.4);
}

/* === LINKS === */
a {
  color: #2c5530;
  text-decoration: none;
  transition: color 0.2s ease;
}

a:hover {
  color: #88d8c0;
  text-decoration: underline;
}

/* === REGISTER LINK (optional) === */
.register-link {
  text-align: center;
  margin-top: 35px;
  color: #666;
}

```

```
    font-size: 15px;
}

.register-link a {
    font-weight: 600;
}

/* === FLASH MESSAGES === */
.flashes {
    list-style: none;
    margin-bottom: 25px;
}

.flashes li {
    background: linear-gradient(135deg, #ffea7 0%, #fab1a0 100%);
    color: #2d3436;
    padding: 15px 18px;
    border-radius: 10px;
    margin-bottom: 15px;
    border-left: 4px solid #e17055;
}

/* === LISTS === */
ul {
    padding-left: 20px;
    margin-bottom: 20px;
}

ul li {
    margin-bottom: 10px;
    font-size: 16px;
}

/* === FOOTER === */
footer {
    background: linear-gradient(135deg, #74b9ff 0%, #0984e3 100%);
    color: white;
    text-align: center;
    padding: 25px 0;
    margin-top: auto;
}
```

```

footer p {
  margin: 8px 0;
  font-size: 14px;
}

/* === RESPONSIVE === */
@media (max-width: 768px) {
  header h1 {
    font-size: 2em;
  }

  .content-wrapper {
    padding: 20px;
    margin: 15px;
  }

  h1, h2 {
    font-size: 1.5em;
  }
}

```

Was wurde verbessert?

Bereich	Änderung
DRY-Prinzip	<code>textarea</code> mit in Input-Styling integriert
<code>main</code>	Kein zentriertes Flex mehr – flexibler für unterschiedliche Inhalte
<code>.content-wrapper</code>	Zentrale Begrenzung für alle Inhalte
Buttons	Für alle Buttons, nicht nur <code>type="submit"</code>
Responsivität	Breitere Media Query (768px), Schriftgrößen angepasst
Optik	Rundere Kanten, weichere Farben, sauberere Trennung von Layout/Design

Wenn du möchtest, kann ich dir zusätzlich eine `dark mode`-Variante oder eine leicht minimalistischere Version erzeugen. Sag einfach, was du brauchst!