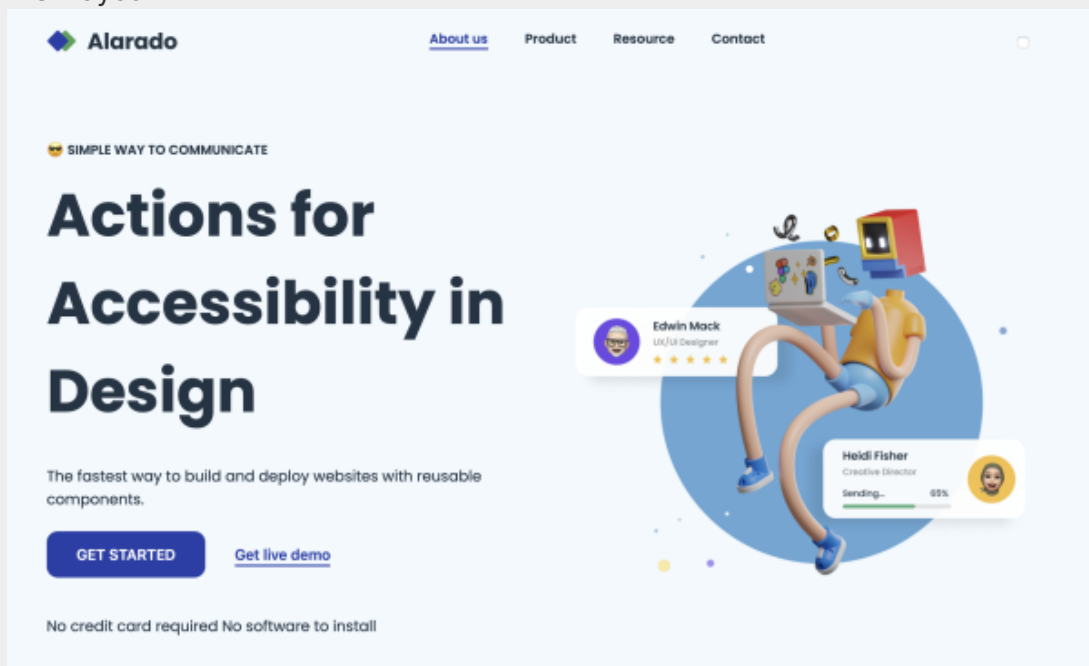


# Projekt 1: Landingpage Alarado – Fonts, Variablen und Responsive

**Ziel:** Sie ergänzen im bestehenden LU01-Projekt die **TODO in LU02:**

1. Google Font einbinden
2. CSS-Variablen in :root definieren und verwenden
3. Typografie, Navigation und Buttons gemäss Design stylen
4. Responsive Anpassungen mit einer Media Query umsetzen

Ziel-Layout:

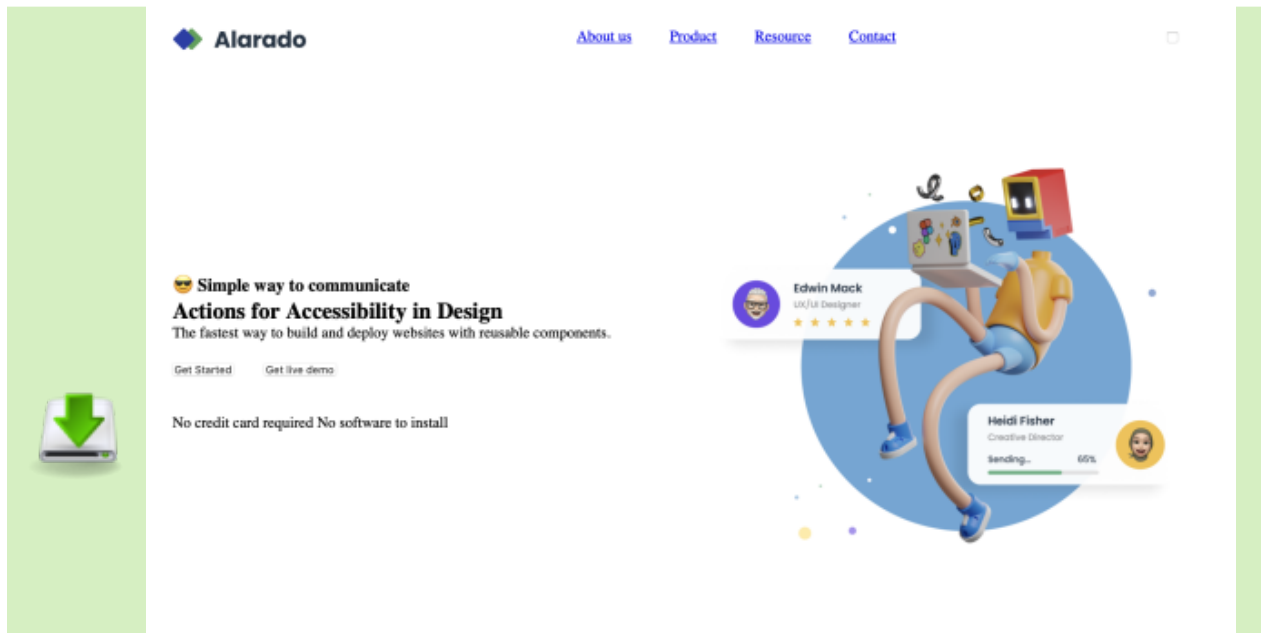


## Voraussetzungen

- Sie arbeiten im bereitgestellten `index.html` Template aus LU01 weiter.

Ausgangslage:





Falls Sie in der letzten Unterrichtseinheit nicht weit gekommen sind, laden Sie sich dieses HTML-File herunter und verbessern Sie Ihren Code nach dieser Vorlage.

Code nach LU01

## Arbeitsweise

- Arbeiten Sie von oben nach unten im Template.
- Nach jedem Schritt: speichern → Browser aktualisieren → Ergebnis prüfen. (oder mit Live-Server – aktualisiert automatisch)
- Wenn etwas nicht funktioniert: DevTools öffnen → Console und Elements prüfen.

### Hilfreiche Links



- Google Fonts: [fonts.google.com](https://fonts.google.com)
- MDN: CSS Custom Properties: [MDN CSS Variables](#)
- MDN: Media Queries: [MDN Media Queries](#)
- DevTools Responsive Mode: [Chrome DevTools Device Mode](#)

## Schritt 1: Google Font Poppins einbinden

### Aufgabe

- Öffnen Sie [Google Fonts](#) und suchen Sie **Poppins**.

- Klicken Sie auf *Get Font*
- Kopieren Sie den vorgeschlagenen `<link>` Code und fügen Sie ihn im `<head>` an der Stelle **TODO in LU02** ein.
- Setzen Sie im body (CSS) die `font-family` auf Poppins mit einem Fallback auf *sans-serif*

## Schritt 2: CSS-Variablen in :root definieren

### Aufgabe

- Füllen Sie den Block `:root` im CSS.
- Definieren Sie Variablen für:
  1. Farben für Hintergrund und Text
  2. Aktive Linkfarbe
  3. Schriftgrößen und Schriftschnitte (=font-weight) für Navigation, h1, h3, Buttons
- Werte nehmen Sie aus dem Figma-Design.
- Nutzen Sie rem-Werte und nicht px-Werte für die Schriftgrößen. Einen Umrechner finden Sie hier: [PX To REM converter](#) oder via VS-Code Extension [VS Code Extension px-to-rem](#) → Umrechnung: pixel wert markieren und dann Alt-Z drücken.

#### Beispielstruktur für :root

```
:root {  
  /* Farben */  
  --color-bg-light: ...;  
  --color-text-light: ...;  
  --color-accent: ...;  
  
  /* Typografie */  
  --size-h1-desktop: ...;  
  --size-h1-mobile: ...;  
  --weight-h1: ...;  
  
  --size-h3: ...;  
  --weight-h3: ...;  
  
  --size-nav: ...;  
  --weight-nav: ...;  
  
  --size-button: ...;  
  --weight-button: ...;  
}
```



**Regel:** Ab jetzt verwenden Sie im CSS möglichst oft `var(--...)` statt einzelne Zahlen zu kopieren.

## Schritt 3: Grundstyles im body setzen

### Aufgabe

- Setzen Sie im body:
  1. background-color und color über Ihre Variablen

#### Beispiel welche Eigenschaften erwartet werden

```
body {  
  font-family: ...;  
  background-color: var(...);  
  color: var(...);  
  height: 100vh;  
}
```

### Überprüfen Sie Folgendes:

- Text wirkt sichtbar in der neuen Schrift.
- Hintergrund und Textfarbe stimmen mit dem Design überein.

## Schritt 4: Navigation stylen



### Aufgabe

- Stylen Sie die Navigation so, dass:
  1. die `li` eine definierte Schriftgrösse und ein Schriftschnitt (=font-weight) erhalten (über Variablen)
  2. Links standardmässig keine Unterstreichung haben
  3. der aktive Link als aktiv erkennbar ist (Farbe und Unterstreichung gemäss Design)
- Achten Sie darauf, dass nur der aktive Link speziell gestylt ist.



Für „alle Links ausser active“ eignet sich ein Selektor wie `a:not(.active)`.<sup>1)</sup>

## Schritt 5: Typografie für h3 und h1

### Aufgabe

- Setzen Sie bei h3:
  1. Schriftgrösse und *font-weight* über Variablen
  2. Grossschreibung mit `text-transform: uppercase`
  3. passende Abstände (margin)
- Setzen Sie bei h1:
  1. Schriftgrösse und *font-weight* über Variablen
  2. passende Abstände gemäss Design

#### Beispiel welche Eigenschaften erwartet werden

```
h3 {  
  font-size: var(...);  
  font-weight: var(...);  
  text-transform: uppercase;  
  margin-bottom: ...;  
}  
  
h1 {  
  font-size: var(...);  
  font-weight: var(...);  
  margin-bottom: ...;  
}
```

## Schritt 6: Buttons stylen



### Aufgabe

- Definieren Sie Basis-Styles für alle button:
  1. Schriftgrösse und *font-weight* über Variablen
  2. Cursor-Verhalten
- Stylen Sie anschliessend die Varianten:
  1. `.highlight` als primärer Button (Padding, Hintergrundfarbe, Radius, Textfarbe)
  2. `.active` als sekundärer Button (transparent, keine Border)

### Beispiel welche Bereiche Sie umsetzen sollen

```
button { ... }  
button:hover { ... }  
  
button.highlight { ... }  
button.active { ... }
```

### Überprüfen Sie folgendes

- „Get Started“ wirkt wie ein Hauptbutton.
- „Get live demo“ wirkt eher wie ein Link.

## Schritt 7: Responsive Regeln mit Media Query



## Aufgabe

- Ergänzen Sie im Template den @media Block so, dass das Mobile-Layout funktioniert:
  1. Hero wird einspaltig
  2. Bild kommt oben, Text darunter
  3. Innenabstände werden kleiner

4. Navigation-Liste und Switch verschwinden
5. Hamburger-Icon wird sichtbar
6. h1 wird kleiner

### Gerüst für den Media Query Block

```
@media screen and (width < 1024px) {  
  /* Hero Layout anpassen */  
  /* Content-Wrapper Breite anpassen */  
  /* Container Padding anpassen */  
  /* Typografie Mobile anpassen */  
  /* Navigation/Switch ausblenden, Hamburger einblenden */  
}
```

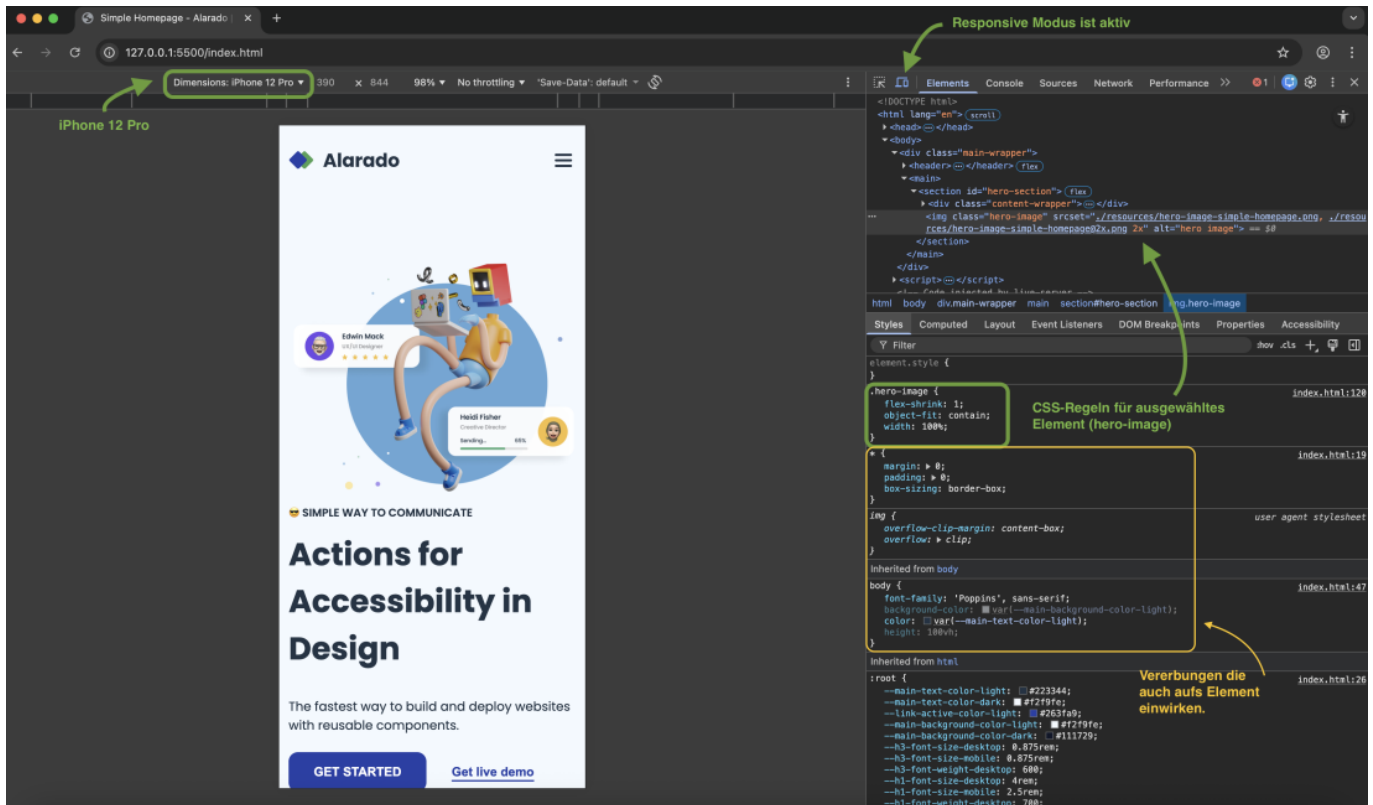


**Kaskade merken:** Wenn ein Selektor im @media Block gleich ist wie oben, überschreibt die Regel im @media Block die vorherige Regel, sobald die Bedingung stimmt.

### Überprüfen Sie Folgendes:

- Über 1024px: 2-Spalten Layout.
- Unter 1024px: 1-Spalten Layout mit Bild oben.
- Hamburger sichtbar, Menu und Switch versteckt.

## Schritt 8: Responsive testen mit DevTools



## Aufgabe

- Öffnen Sie DevTools und aktivieren Sie den Device Toolbar Modus.
- Testen Sie mindestens:
  1. Breite knapp über 1024px
  2. Breite knapp unter 1024px
  3. ein Smartphone-Preset (z.B. iPhone)

## Achten Sie besonders auf Folgendes:

- Kein horizontales Scrollen.
- Keine überlappenden Elemente.
- Text bleibt gut lesbar.

1)

Der Selektor trifft alle Links, die nicht die Klasse active haben.

From:

<https://wiki.bzz.ch/> - BZZ - Modulwiki

Permanent link:

[https://wiki.bzz.ch/de/modul/m291/learningunits/lu02/aufgaben/a\\_projekt\\_styles](https://wiki.bzz.ch/de/modul/m291/learningunits/lu02/aufgaben/a_projekt_styles)

Last update: 2026/02/02 02:19

