**Development Reference Guide**

**📂 Project Structure**

* **HTML Files**
  + index.html: Main HTML file structuring the content.
* **CSS Files**
  + /CSS/
    - style.css: Main stylesheet for the project.
    - /CSS/boilerplate/: Contains boilerplate styles (e.g., reset.css, base.css).
* **Assets**
  + /assets/: Contains images, icons, and other media files.
    - logo-dark.svg
    - logo-light.svg
    - image-hero-desktop@2x.webp
    - image-hero-tablet@2x.webp
    - image-hero-mobile@2x.webp
    - Other icon files (e.g., icon-animation.svg, icon-design.svg, etc.)

**🎨 CSS Organization**

**1. Separate Folder for Boilerplate Styles**

* **Boilerplate Styles**: Create a dedicated folder (e.g., /CSS/boilerplate/) to store essential styles like CSS resets and base styles.
* **Example Files**:
  + reset.css: Resets default browser styles.
  + base.css: Contains base styles for typography, colors, and common elements.

**2. BEM Naming Convention**

* **Block**: Represents the higher-level component (e.g., header).
* **Element**: A part of the block, separated by \_\_ (e.g., header\_\_logo).
* **Modifier**: A variant or state of a block or element, separated by -- (e.g., button--dark).
* **Multiple Classes**: Classes can be combined for more specific styling (e.g., button dark).

**Example:**

<header class="header wrapper">

<img class="header\_\_logo" src="/assets/logo-dark.svg" alt="Skilled Home" />

<a href="#" class="button button--dark">Get Started</a>

</header>

## 🎯 Best Practices

### 1. ****Responsive Design****

* **Media Queries**: Use @media rules to adjust styles for different screen sizes.
* **Picture Tag**: Utilize the <picture> element to serve different image sizes based on device pixel ratio (DPR) and viewport size.

**Example:**

<picture>

<source media="(min-width: 75em)" srcset="/assets/image-hero-desktop@2x.webp" width="2092" height="1876">

<source media="(min-width: 50.625em)" srcset="/assets/image-hero-tablet@2x.webp" width="695" height="723">

<img src="/assets/image-hero-mobile@2x.webp" alt="Description" width="870" height="818">

</picture>

**Fluid Typography**: Use the clamp() function to create fluid font sizes that adjust between a minimum and maximum value without media queries.

**Example:**

h1 {

font-size: clamp(1.5rem, 2.5vw, 2.5rem);

}

* **Responsive Design Tools**:
  + **Browser Inspect Tools**: Use the device toolbar to toggle different device views.
  + **Custom Resolutions**: For laptops, use resolutions like 1366x768.
  + **Network Tab**: Check image rendering and loading performance.

**2. CSS Layouts**

* **Flexbox**: Ideal for aligning items when column sizes are flexible.
  + **Properties**:
    - justify-content: Aligns items horizontally.
    - align-items: Aligns items vertically.
* **Absolute Positioning**:
  + **Parent**: Set to position: relative;.
  + **Child**: Set to position: absolute; to remove from normal flow and not take up space.

**Example:**

.parent {

position: relative;

}

.child {

position: absolute;

top: 0;

left: 0;

}

.parent {

position: relative;

}

.child {

position: absolute;

top: 0;

left: 0;

}

### 3. ****Typography****

* **Font Sizing**:
  + Convert px to rem or em by dividing by 16 (e.g., 32px = 2rem).
* **Font Sources**:
  + **Google Fonts**: Not GDPR compliant. Use alternatives like Google Webfont Helper to self-host fonts.

### 4. ****Colors****

* Define a color palette in your CSS or a separate variables file for consistency.

### 5. ****Accessibility (A11y)****

* **Alt Attributes**:
  + **Descriptive Images**: Provide meaningful alt text.
  + **Non-Descriptive Images**: Use empty alt="" to prevent screen readers from reading file paths.

### 6. ****Performance****

* **Device Pixel Ratio (DPR)**: Use 2x images for higher-resolution displays to ensure clarity.
* **SVGs**: Prefer SVGs for icons and simple graphics for better scalability and performance.
* **CDNs**: Utilize Content Delivery Networks to serve assets faster, but ensure GDPR compliance.

### 7. ****Code Readability****

* **Line Length**: Keep lines between 50 to 60 characters for better readability.
* **Class Naming**: Use multiple classes separated by spaces for modularity.

## 🛠️ Tools and Resources

* **Browser Compatibility**: Use [Can I Use](https://caniuse.com/) to check browser support for CSS features.
* **Fluid Typography Calculator**: [Royal Fig's Fluid Typography Calculator](https://royalfig.github.io/fluid-typography-calculator/) to generate clamp() values.
* **Google Webfont Helper**: Google Webfont Helper for self-hosting Google Fonts.
* **Responsive Design Testing**: Utilize browser developer tools' responsive design features to test various device viewports.

## 📐 Design Specifications

* **Device Dimensions**:
  + **Phone**: 375x812 (e.g., iPhone X)
  + **Tablet**: 810x1080
  + **Laptop**: 1200x900

## 🔄 Transitions and Animations

* **Transitions**:
  + Apply transitions to specific properties that need animation instead of the entire element for better performance.

**Example:**

.button {

transition: background-color 0.3s ease;

}

**🖼️ Image Handling**

* **Picture Tag Usage**:
  + Serve different image sizes based on screen size and DPR.
  + Provide a fallback/default image for unsupported browsers.
* **Image Formats**:
  + Prefer webp for better compression.
  + Use svg for vector-based graphics and icons.

**📏 Layout Units**

* **REM vs. EM**:
  + Both rem and em are relative units, but rem is relative to the root (html) font size, while em is relative to the parent element's font size.
* **Inline vs. Block**:
  + **Inline Elements**: Align horizontally (left and right).
  + **Block Elements**: Stack vertically (top and bottom).

**🛡️ GDPR Compliance**

* **Fonts**: Self-host fonts to comply with GDPR instead of relying on external sources like Google Fonts.
* **Data Handling**: Ensure that any user data collected adheres to GDPR rules if targeting EU users.

**📚 Additional Tips**

* **HTML5 Void Elements**: Do not use self-closing tags for void elements (e.g., <img>, <br>, <hr>, etc.).

**Incorrect:**

<img src="image.png" alt="" />

Correct:

<img src="image.png" alt="">

* **Padding and Spacing**: Use consistent padding values to maintain layout harmony.
* **Class Naming**: Use multiple class names for modular and reusable styles (e.g., class="button dark").