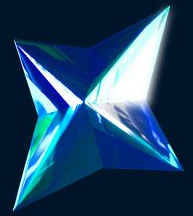
BRO Academy



Preparation for the Final Project

Lesson plan

1
Review of the criteria

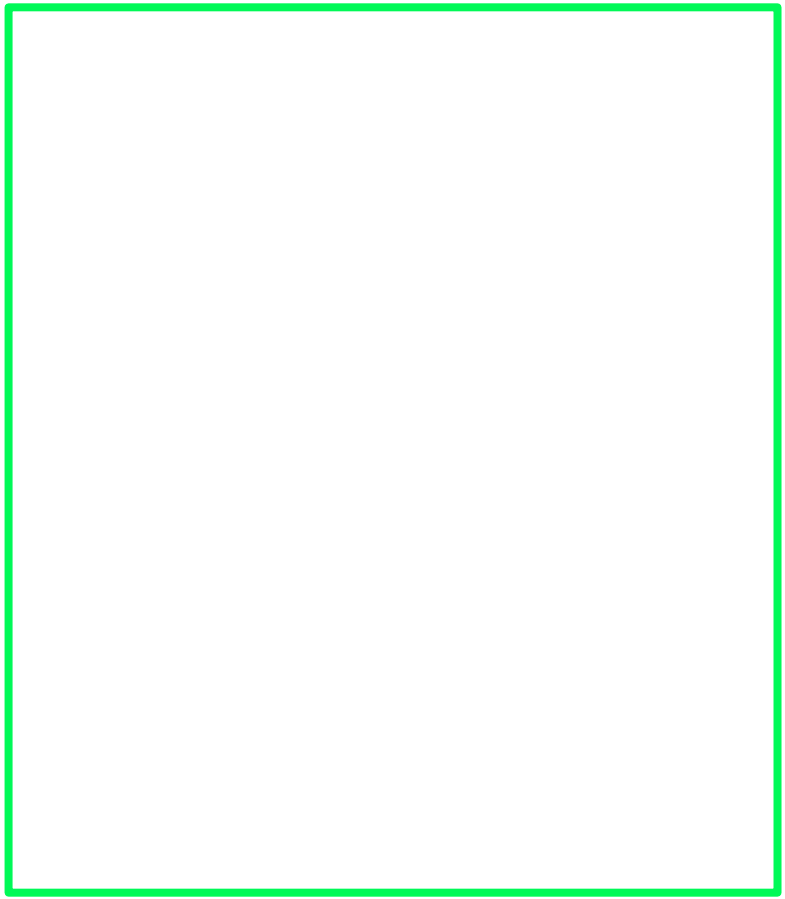
2
CSS optimization

3
Content overflow

4
Pixel perfect

5
Testing on different devices

Quality criteria for coding



WHY?

- For developing websites according to the latest industry standards.
- For optimizing the website to ensure fast loading.
- For user-friendly website experience.

Course completion criteria

- To successfully complete the course, the project must meet **all mandatory requirements** ❤️
- Basic criteria – 💛 will allow you to defend the project with a **maximum grade**
- Bonus criteria – 💚 if you've done everything and want to try something extra
- There will be **three iterations** (attempts) to submit the project.

Criteria

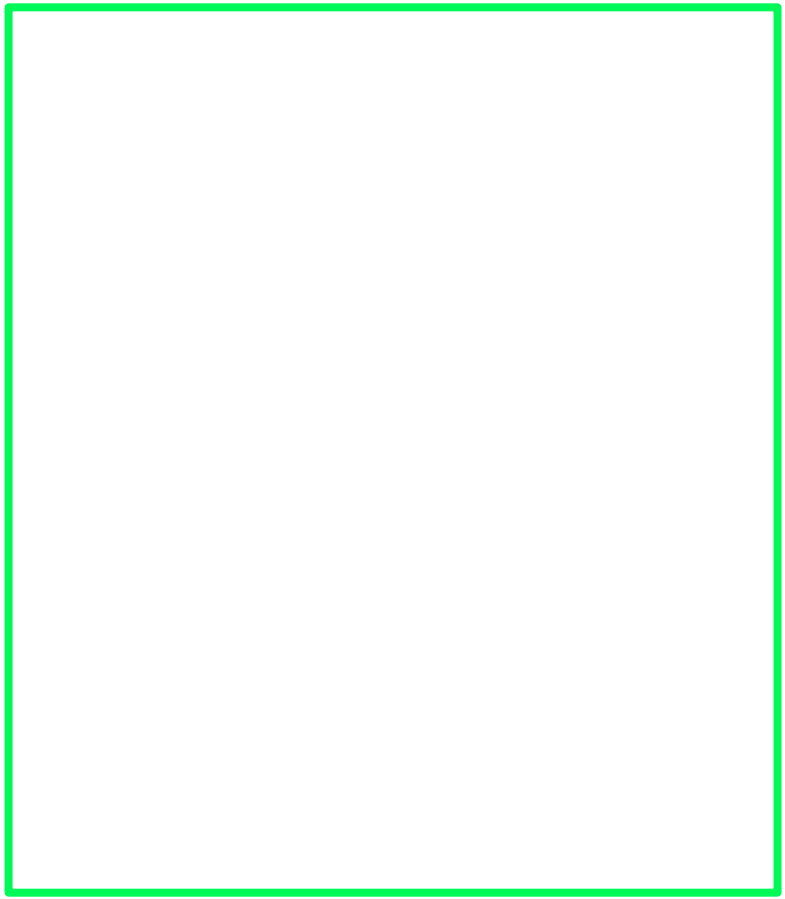
✅ HTML

✅ CSS

✅ Images

✅ Layout

👀 For all site



CSS Optimization

Why?

The website loads faster →
→ easier for users to navigate
→ they spend more time on the site
→ higher chances of purchase

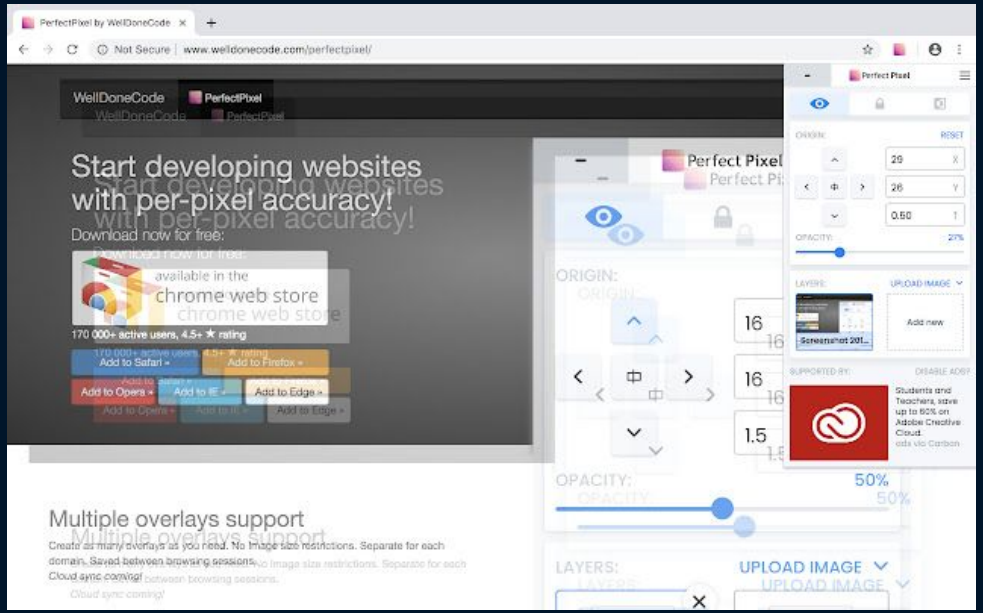
How?

- CSS minification (clean-CSS, CSSNano)
- Removing unused CSS (PurifyCSS, UnCSS)
- Using shorthand properties (e.g., margin, padding, border, etc.)
- Combining selectors that have the same styles
- Avoiding specificity (no more than two levels of nesting)

Perfect Pixel

Matching the layout to the design pixel by pixel.

Blocks must be positioned exactly 1:1 as in the design. A text margin difference of **up to 5px** is allowed. Adjusting the sizes and positioning of poorly designed blocks is acceptable (differences of 1-2px across different pages are allowed).



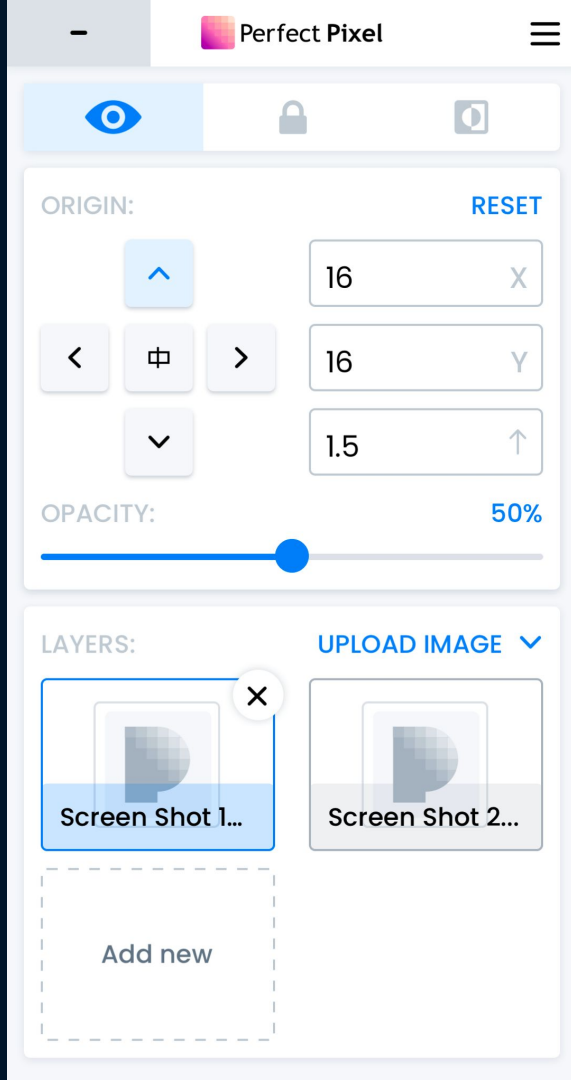
Why is it important:

The client receives work that matches the approved design.

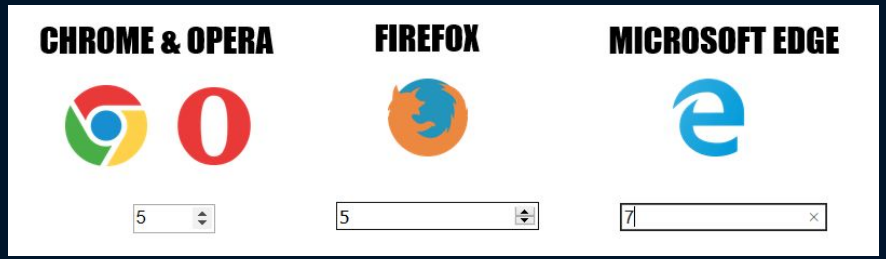
How to Achieve Pixel Perfect

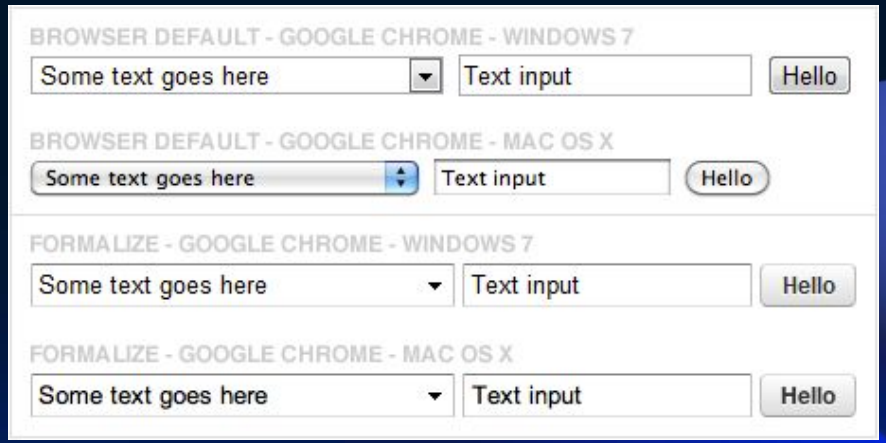
1. Download the Perfect pixel plugin for Chrome.
2. Upload the design screenshot.
3. Set the screen width to match the screenshot.
4. Set coordinates to 0 0.
5. Make the screenshot semi-transparent.
6. Click the lock icon to fix the screen, so it doesn't move when scrolling.
7. Open DevTools and adjust the necessary margins/padding/sizes.
8. Transfer these values to VS Code.

Do not change the screenshot's scale.
You can use the inversion mode if it helps.



Consistent Display in Browsers





The last two versions of browsers:
Chrome, Opera, Firefox, Safari, Edge

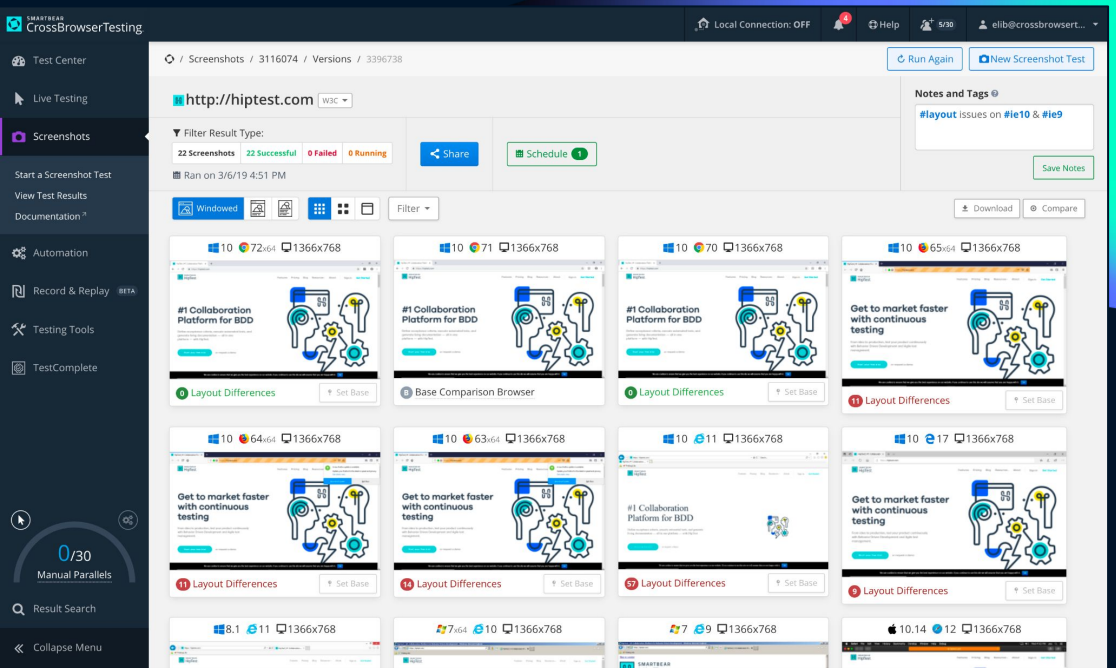
All browsers have different rendering methods, and some elements may look different.

A difference of 2-3 pixels between browsers is normal.

The main thing is that the grid should remain consistent.

How to Achieve Cross-Browser Consistency

- Install all available browsers on your computer, smartphone, and tablet.
- Use testing services for specific devices (some are partially free).



[BrowserShots.org]

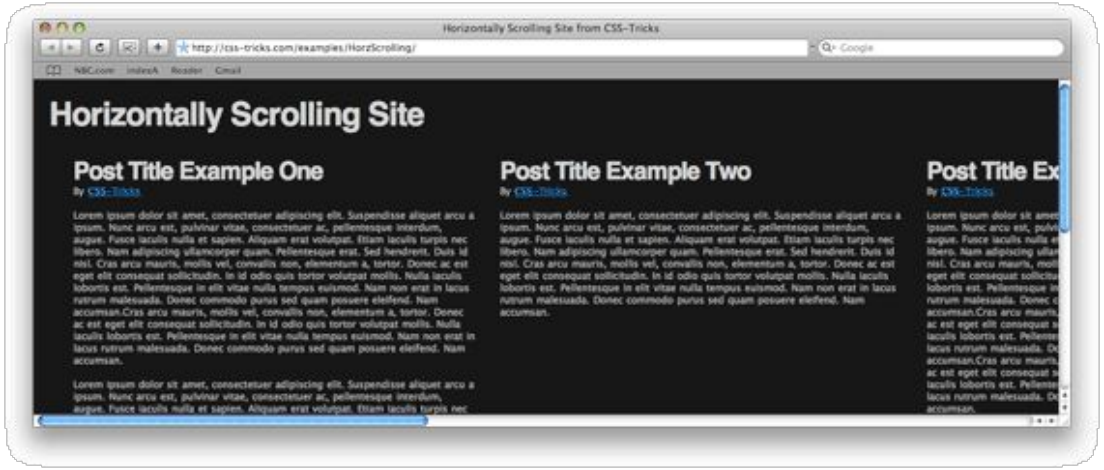
[BrowserStack]

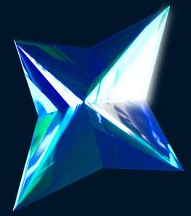
[LambdaTest]

[CrossBrowserTesting]

Horizontal Page Scrolling

- **Add outline to all blocks** (*) and see which one goes beyond the screen boundaries.
- **Remove blocks** one by one and check after which block's removal the scroll disappears.
- Use the "**overflow**" indicator in Firefox.
- **Remove 100vw** (on Windows, scrollbar width is added to this width).





Content Overflow

Why?

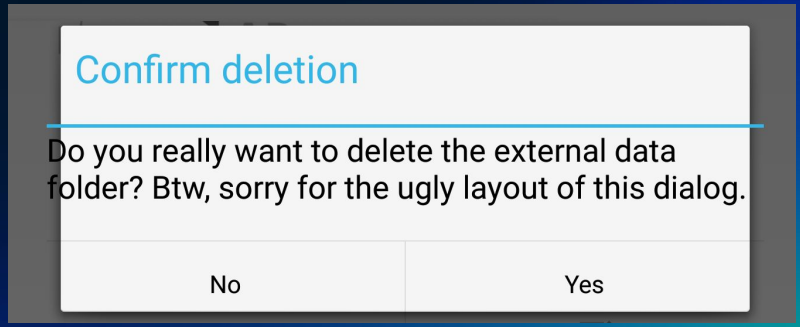
After development, **anything can change**: text/images might be added, removed, or updated. Websites are often connected to a CMS.

Content Overflow

It is time to
paaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaarty

How to check:

– **Replace images** with ones larger/smaller than the original.
– Add **more/less text** wherever it’s present.
– Insert **long words**.



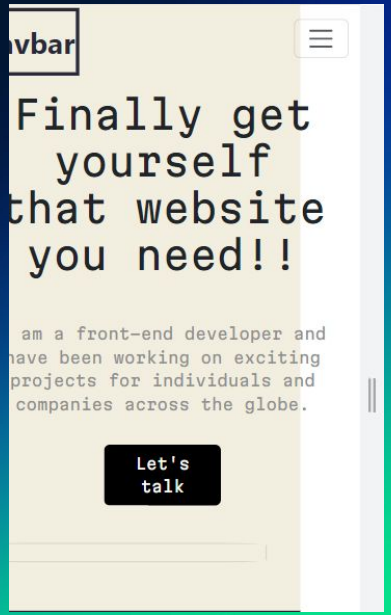
The layout should not break, and text should not overflow beyond the block's boundaries.

Fixing Content Overflow

- Avoid using fixed width and height,
  only use minimum values if needed.
- Use `overflow` to hide text that doesn't fit or to create scrollbars within a block.
- If you need to cut text:

```
overflow: hidden;
white-space: nowrap;
text-overflow: ellipsis;
```

- For long words in small fields, use properties like `word-break` or `overflow-wrap`
- Use `flexbox` to allow wrapping
- Set a maximum block width of 100%.
- Avoid using negative margins.



Without overflow-wrap

This is a test to see how the overflow wrap works in CSS. This longwordcantfit

```
.card {
  /* without wrapping */
}
```

With overflow-wrap

This is a test to see how the overflow wrap works in CSS. This longword cantfit

```
.card {
  overflow-wrap: break-word;
}
```

Without hyphens

This is a test to see how the overflow wrap works in CSS. This longwordcantfit

```
.card {
  /* without hyphens */
}
```

With hyphens

This is a test to see how the overflow wrap works in CSS. This longwordcan-tfit

```
.card {
  hyphens: auto;
}
```

Without hyphens

This is a test test to see how the hyphens cool property will handle different word lengths.

```
.card {
  /* without hyphens */
}
```

With hyphens

This is a test test to see how the hy-phens cool property will handle different word lengths.

```
.card {
  hyphens: auto;
}
```

Without truncation

This is a cool test for CSS truncation.

```
.card {
  /* without hyphens */
}
```

With truncation

This is a cool test fo…

```
.card p {
  white-space: nowrap;
  overflow: hidden;
  text-overflow: ellipsis;
}
```

Single line truncation

This is a cool test fo…

```
.card p {
  white-space: nowrap;
  overflow: hidden;
  text-overflow: ellipsis;
}
```

Multiline truncation

This is a cool test for multiline truncation in CSS. It's a useful…

```
.card p {
  display: -webkit-box;
  -webkit-line-clamp: 3;
  -webkit-box-orient: vertical;
  overflow: hidden;
}
```

BRO
Academy