In above example, we have used three classes from Tailwind utilities. Each one is described in below.

* **Background color:** We used bg-grey-lighter to give light grey color to the background. The syntax for this is bg-{color}. Where color can be anything from values mentioned in [background docs](https://tailwindcss.com/docs/background-color) and [color docs](https://tailwindcss.com/docs/colors/).
* **Screen Height:** Class h-screen will make the element height same as screen height, it’s 100vh in CSS. You can use the h-{size} syntax to use other values as well.
* **Font Family:** This class sets our body font family to sans family of fonts.
* **Container:** This utility (container) works same as Bootstrap’s container, setting up max-width of the page. The only difference is that Tailwind doesn’t add margin’s automatically to left and right of the element to center it on the page. Therefore, we need to add mx-auto to center the element.
* **Height 100%:** We use h-full to stretch the height of an element same as that of its parent height, it is equivalent to height: 100%. Refer to official docs of [height](https://tailwindcss.com/docs/height) and [width](https://tailwindcss.com/docs/width) for more details.
* **Flexbox:** I must admit that Tailwind makes it enjoyable to use Flexbox. In our case, we have used flex to specify that we will be using CSS flex display and we have added justify-center and items-center to horizontally and vertically center the child elements inside our div.
* **Elements Widths:** We can use w-{size} syntax to specify the width of our elements. In this case we are using w-1/3 to set the width of our element to one third of its parent element. It’s equivalent to set an element class to col-md-4 col-md-offset-4 in Bootstrap.
* **Text Styling and Positioning:** There are several ways to style text in Tailwind. In above, font-hairline sets the CSS font-weight property to 100 and text-center is equivalent to CSS’s text-align: center.
* **Margins and Paddings:**: This is one of my favorite parts. You no longer need to shift between your HTML and CSS files to set different margin and padding sizes for your elements. In Tailwind, you can easily use p{side?}-{size} and m{side?}-{size} to add margins and paddings. We have used mb-6 which sets the bottom margin of our header to **1.5rem**. Below is a list of possible values we can replace with **size** placeholder in above example.
* **Login Box:** We have given our login box a white background (bg-white) along with shadows (shadow-lg) so it can stand out on our grey colored background page. We have also added a **1.5rem** bottom margin and a padding of **2rem** to all four sides.
* **Borders and Radius:** You can use border{-side?}{-width?} and border-{color} for specifying a border’s width, color and side. rounded{-radius?} is used to set our border’s radius.
* **Labels:** We put a small margin to the bottom of label using mb-2 and also we have used font-bold to set our font’s weight to bold. Both label and input use block utility which is same as display: block in CSS.
* **Appearance None:** If you want to remove and reset all browser default styling on an element, use appearance-none utility. This way you can fully customize the design of the elements you work with.
* **Hover:** If you want to add a specific style to an element on states like hover or focus, you can use hover:{whatever-style-class} syntax. In the above example, we change the text box border to darker grey when you hover on it (hover:border-grey).

**Note:** We don’t have a border width with value **12px** in Tailwind’s config file by default. We have used this to give our login div’s top border a size of **12px** (border-t-12). Please add it manually as shown below and then run the build command again to generate a new dist.css.