

# Setting up a Framework to build Google's Autocomplete

Let's set the template as a framework to begin building Google's Autocomplete Functionality.

## WE'LL COVER THE FOLLOWING ^

- The document structure
- The HTML

## The document structure #

Let's layout the minimum amount of HTML needed to have something to work with as we build out the functionality. Remember, we're focusing on JavaScript first for this component.

The search component looks like it can be split up into three main sub-components:

- The search bar where the input and icon are
- The autocomplete results
- The buttons

The search bar will consist of the input element and something (maybe a `div`, maybe a `button`) for the microphone icon (whatever it does isn't within the scope of this project, but we'll account for its presence nonetheless).

The autocomplete results look like it can just contain an unordered list as the sole element. Unordered in this case means that HTML's `ul`, as opposed to `ol` – ordered list. The name shouldn't be taken too seriously. Its meaning is closer to un-numbered-by-default. The buttons are just two buttons, as pointed out in the scoping phase. We'll have to anticipate that it'll need to blend in with the autocomplete results when they show up. We should be able to manipulate it as required with JavaScript

manipulate it as required with javascript.

## The HTML #



Let's write some code.

Output

HTML

Google search

I'm Feeling Lucky



All this should feel familiar to you after the last lesson.

You might be wondering why there's a `div` for `search__suggestions` that just wraps the `ul`. Why can't we just have `ul` as the `search__suggestions`? In general, I always wrap a `div` around semantic HTML elements (elements that convey purpose with their name, as `ul` does). It usually ends up paying off if something unexpected needs to be added, and browsers usually have custom CSS applied to semantic elements (e.g., some slight margin-left to `ul` on Chrome).

That should be all there is to it. We can start thinking about what happens when a character is inputted in the following chapter.