**Styling React Components**

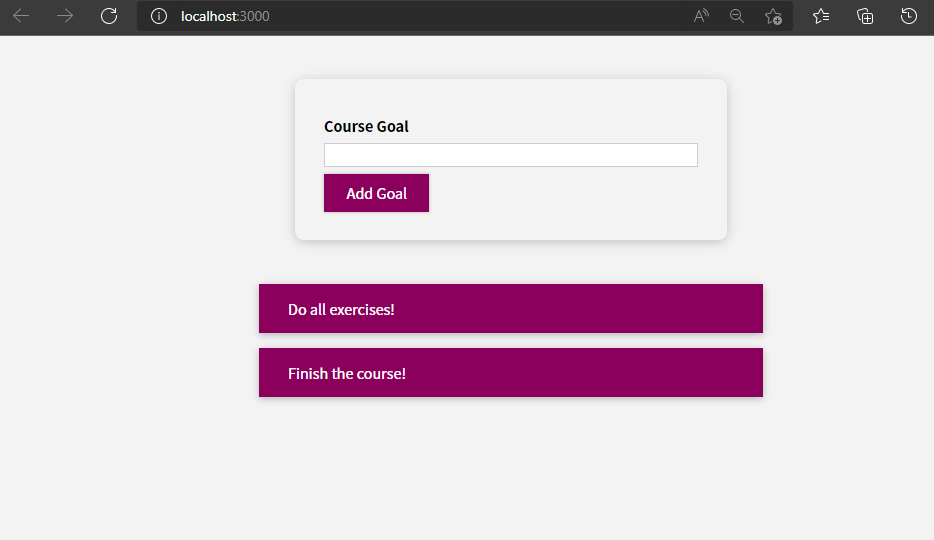
Xây dựng web apps với React không chỉ việc kết hợp các components lại với nhau và đảm bảo các hoạt động logic của nó đúng đắn. Đó được cho là phần quan trọng nhất, phần chủ lực nhất trong React. Tuy nhiên, xây dựng React app còn bao gồm việc định dạng (styling) chúng, để chúng có một cái nhìn đẹp đẽ. Do đó, định dạng cũng là một trong các phần quan trọng trong xây dựng các components.

Ở phần này, chúng ta sẽ tìm hiểu về các cách định dạng cơ bản với điều kiện nhất định. Ví dụ như thay đổi định dạng của một thành phần nào đó khi có một điều kiện nào đó xảy ra. Cụ thể là khi người dùng nhập giá trị lỗi vào input field và bạn muốn input field đó chuyển màu đỏ để báo lỗi.

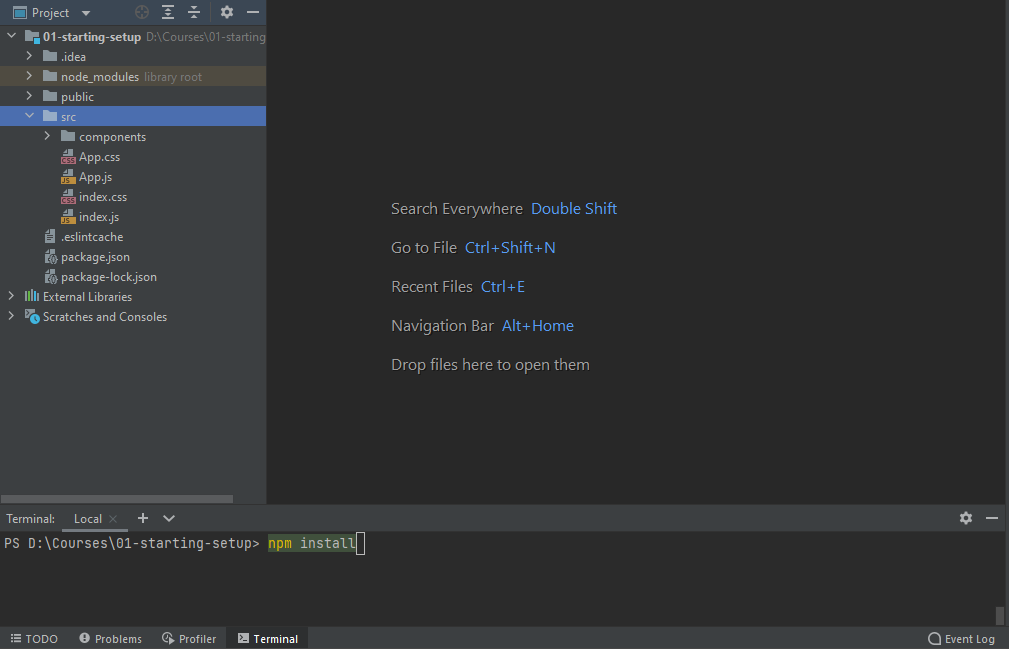
Để bắt đầu phần này, các bạn cần tải về folder project dưới đây.



Một giao diện căn bản được tôi chuẩn bị sẵn trong project này bao gồm chức năng xử lý hiển thị thêm và xóa.



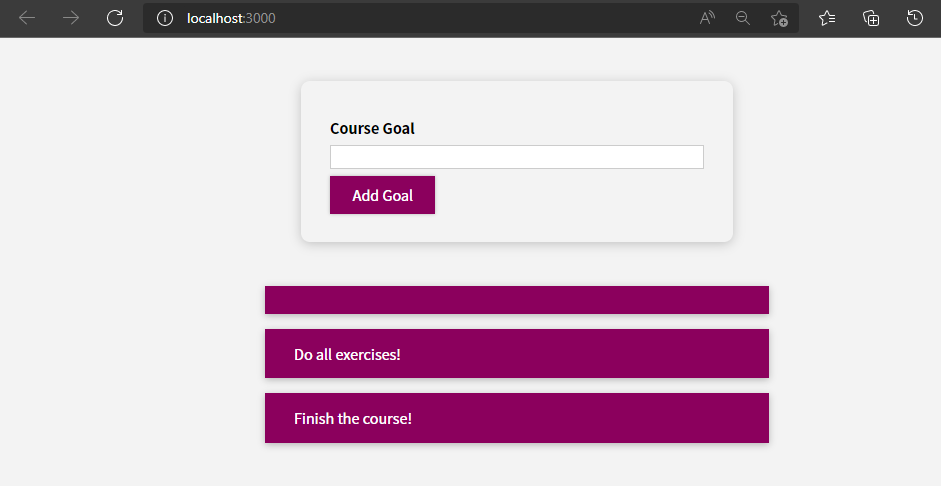
Sau khi tải, các bạn hãy giải nén nó và mở project. Kế tiếp, các bạn cần mở terminal và gõ dòng lệnh “npm install” để thực hiện tải các thư viện bổ sung cần thiết cho phần này.



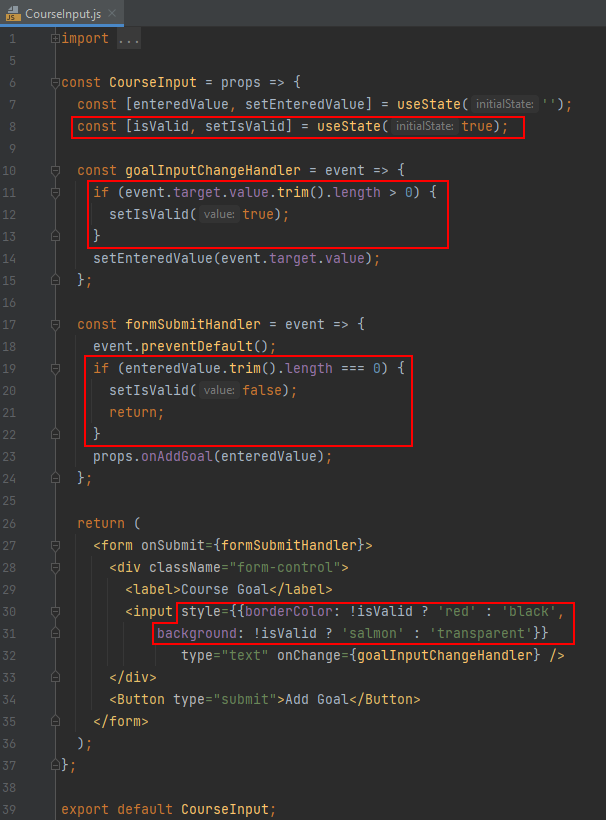
**Dynamic Inline Styles**

Hay chính là các định dạng (style) được viết trực tiếp vào đoạn code JSX trong file js.

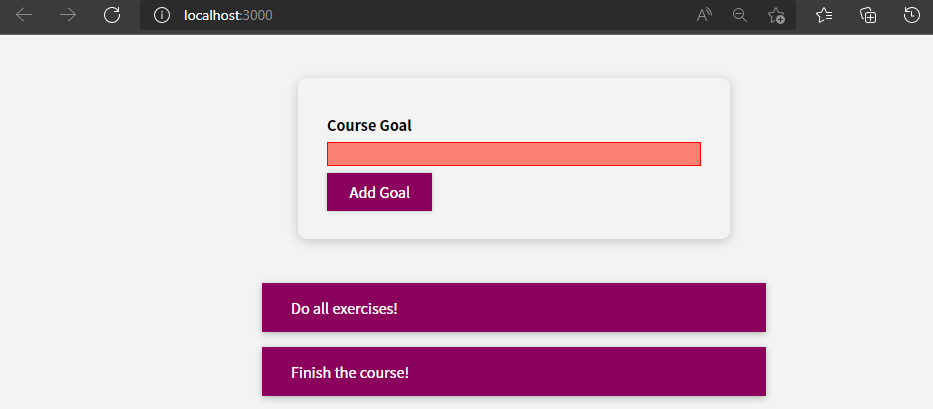
Ví dụ với project có sẵn bên trên. Ở đây ta có một vấn đề xảy ra với giao diện này là khi ta không nhập gì vào input field và bấm Add Goal thì nó vẫn thực hiện thêm vào list phía dưới.



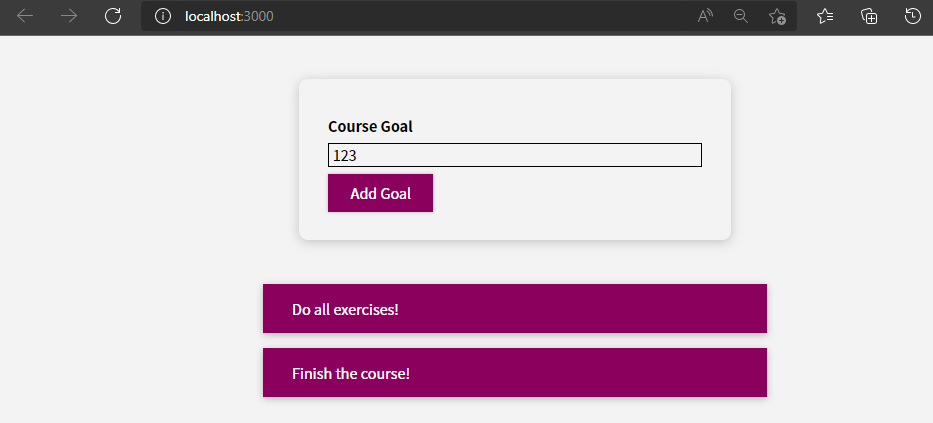
Giờ chúng ta sẽ ngăn việc này xảy ra và hiển thị lỗi một cách trực quan cho người dùng.



Với các đoạn code khoanh đỏ được bổ sung vào, khi chạy lại web ta sẽ được kết quả sau:



Lúc này nếu ta bỏ trống input field và bấm Add Goal thì thanh input field sẽ chuyển sang màu đỏ để báo lỗi.



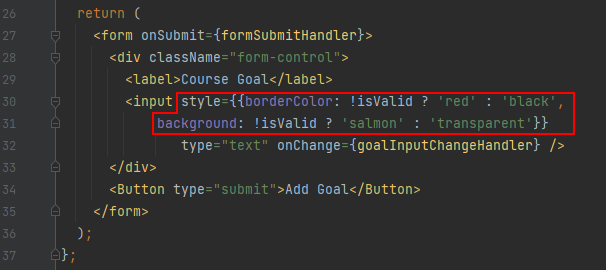
Và nó sẽ trở lại bình thường khi ta nhập vào input field.

**CSS Classes Dynamically**

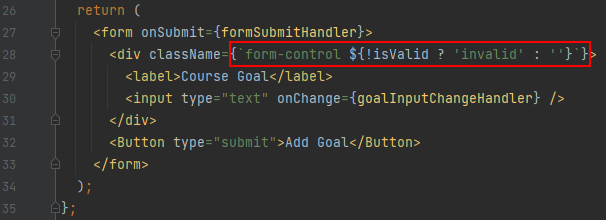
Cũng như cách viết giao diện HTML và định dạng CSS thông thường. Ngoài việc ta có cách chèn trực tiếp đã kể trên, ta còn cách khác chính là định dạng bằng class.

Tiếp tục với ví dụ bên trên:

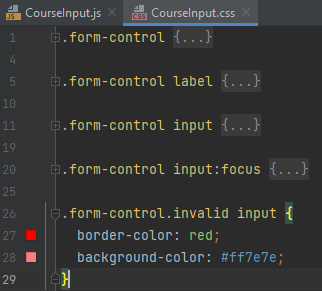
Lúc này ta sẽ bỏ đi đoạn style trực tiếp trong JSX.



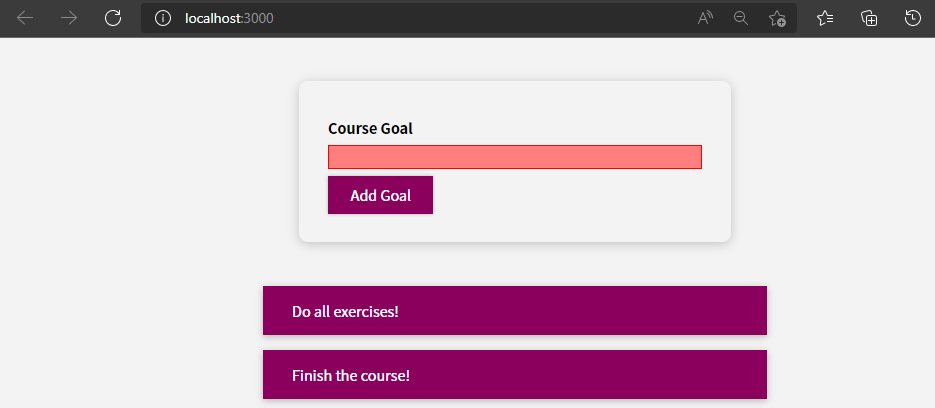
Thay thế bằng một class ‘invalid’ với cú pháp className={`form-control ${!isValid ? ‘invalid’ : ‘’}`}

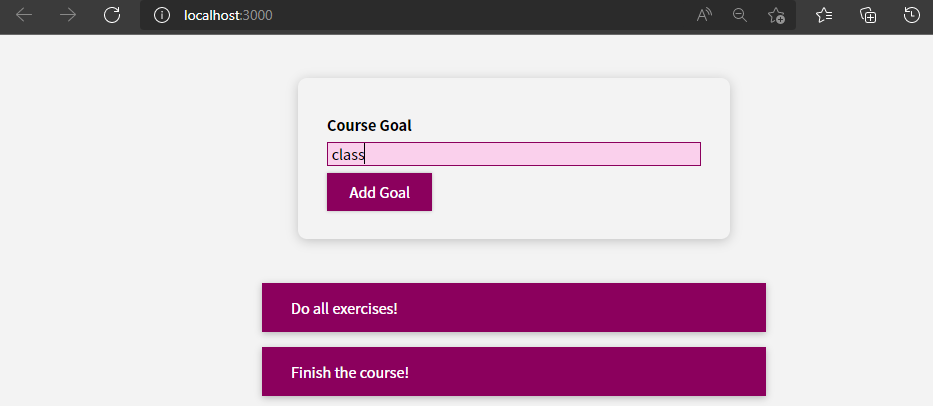


Viết định dạng cho class ‘invalid’ bên file css



Và tất nhiên, kết quả ta nhận được không khác gì so với việc style trực tiếp trong JSX. Tuy nhiên, bằng cách viết style bằng class sẽ giúp phần code chúng ta gọn gàng hơn.

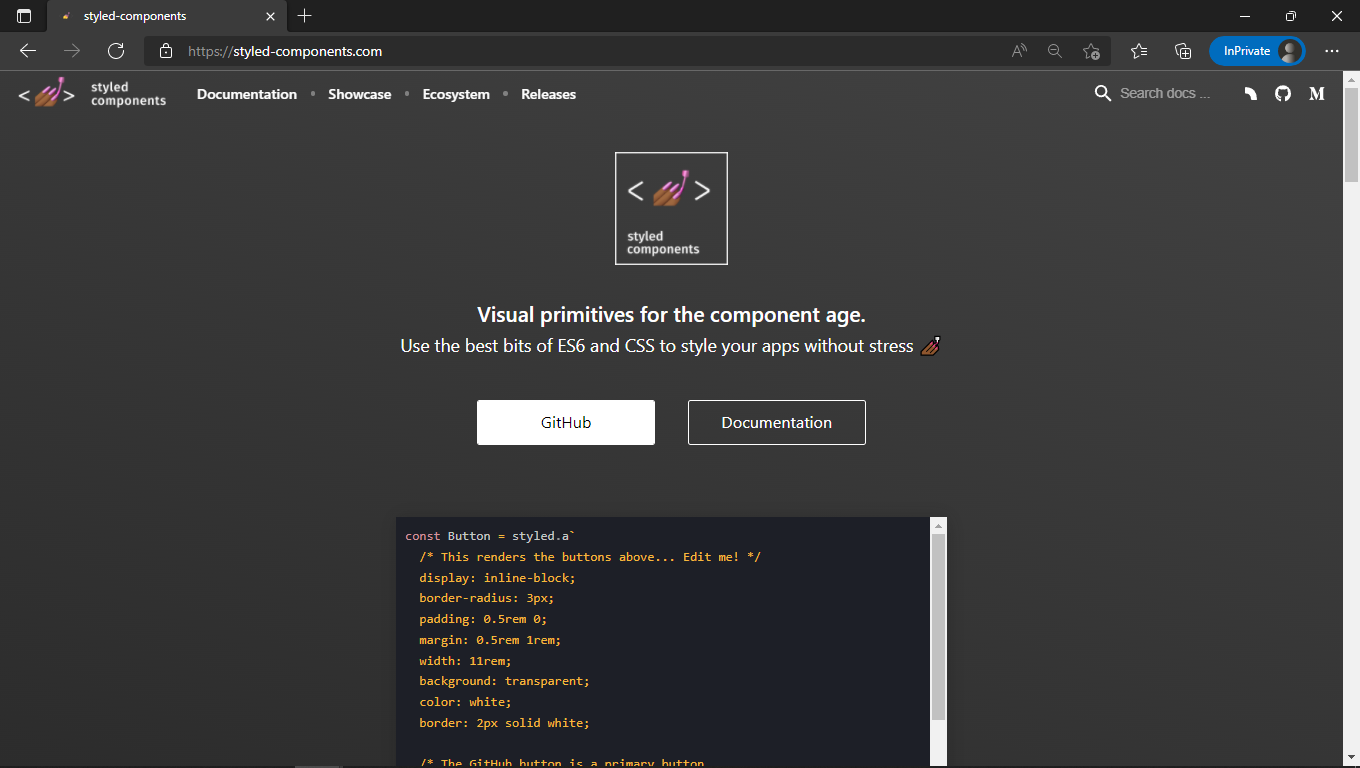




**Styled Components**

Một cách định dạng khác là sử dụng thư viện styled components. Trong phần này, tôi chỉ giới thiệu những thứ cơ bản. Ngoài ra, chúng ta có thể học tất tần tật các thứ liên quan styled components ngay tại trang web của họ.

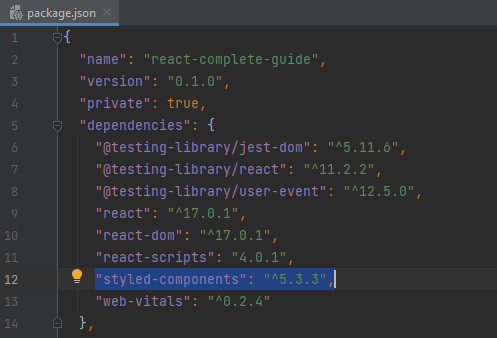
Đường dẫn trang web: <https://styled-components.com>



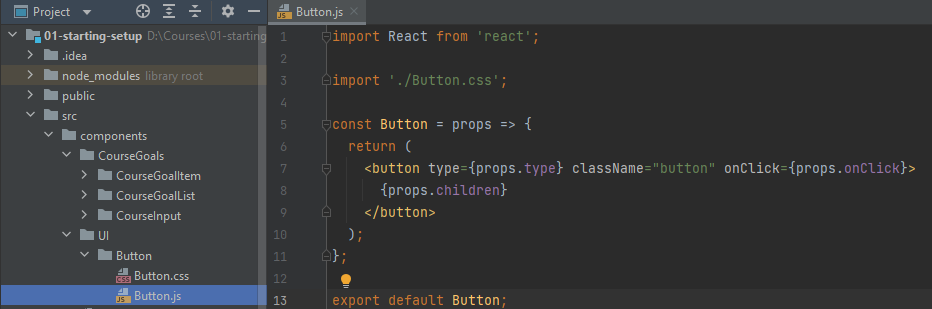
Ta bắt đầu với việc tải thư viện styled-components, gõ và chạy câu lệnh sau trong terminal:

npm install --save styled-components

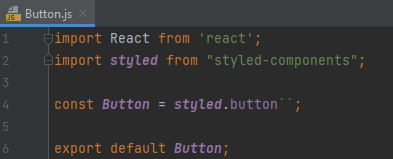
Sau khi tải thành công, một dependency của styled-components sẽ được thêm tự động trong file package.json



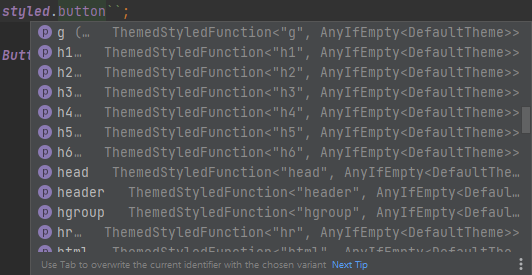
Bắt đầu với component Button trong project, ta thay đổi nó với styled-components.



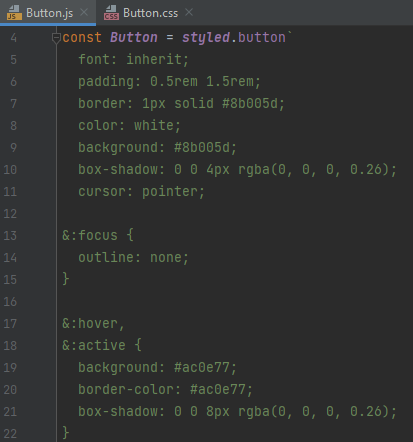
Chúng ta sẽ xóa đi functional Button và import file Button.css. Sau đó chúng ta sẽ import styled và tạo một const Button mới. Lưu ý, const Button này không là một functional component mà là một cú pháp chuẩn để sử dụng styled-components.



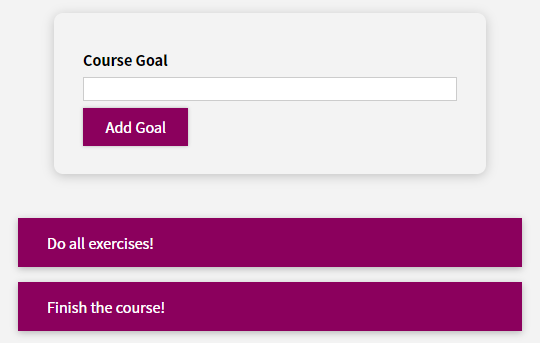
Ở đây chúng ta đang viết lại style cho button vốn có nên sẽ sử dụng phương thức styled.button. Ngoài ra, còn có rất nhiều các phương thức khác.



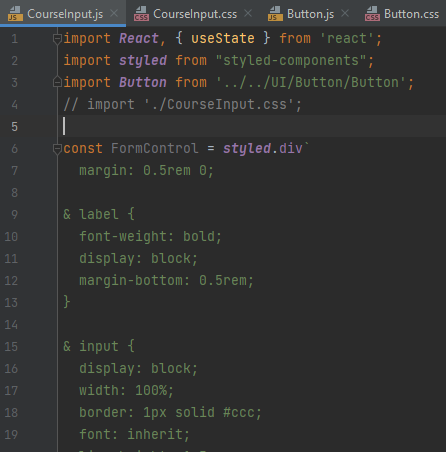
Tiếp theo, ta copy toàn bộ định dạng ở file Button.css vào dấu ` `. Vì đây là một phương thức button nên sẽ không cần đến các selector vốn có trong CSS. Selector của các trường hợp đặc biệt như focus, hover, active,… sẽ được thay thế bằng dấu &.



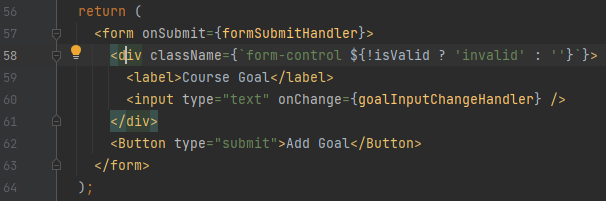
Và cuối cùng ta cũng có kết quả như các cách bên trên

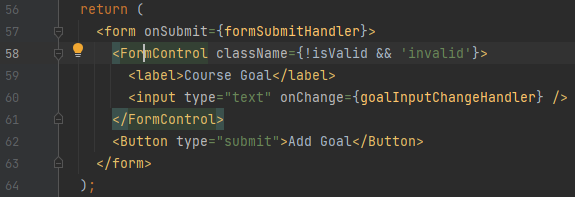


Tương tự với component Button, ta cũng sẽ import styled và đưa toàn bộ CSS ở file CourseInput.css vào const FormControl. Ta sẽ thay thế cho thẻ div vốn có nên ở đây sẽ sử dụng styled.div



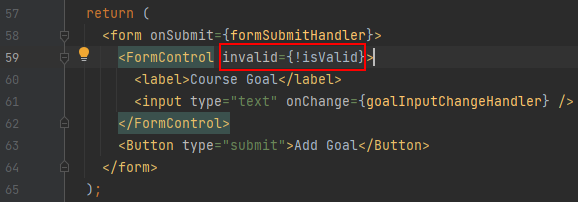
Ta thay thế thẻ div vốn có thành FormControl



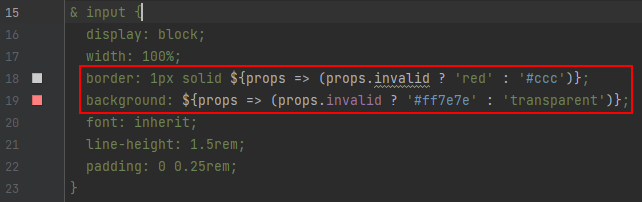


**Props with Styled Components**

Là một phương pháp khác được styled-components hỗ trợ. Với invalid={} sẽ tồn tại hoặc không phụ thuộc tham số true/false trong nó. Ở đây, nếu người dùng bỏ trống field và bấm Add thì isValid = false, tham số invalid nhận được là true và ngược lại.



Với cách props, ta sẽ thêm điều kiện vào đoạn styled CSS bên trên theo tham số invalid.



Sau đó, ta có thể bỏ đi đoạn định dạng này vì props điều kiện thêm bên trên đã thay thế cho nó.

