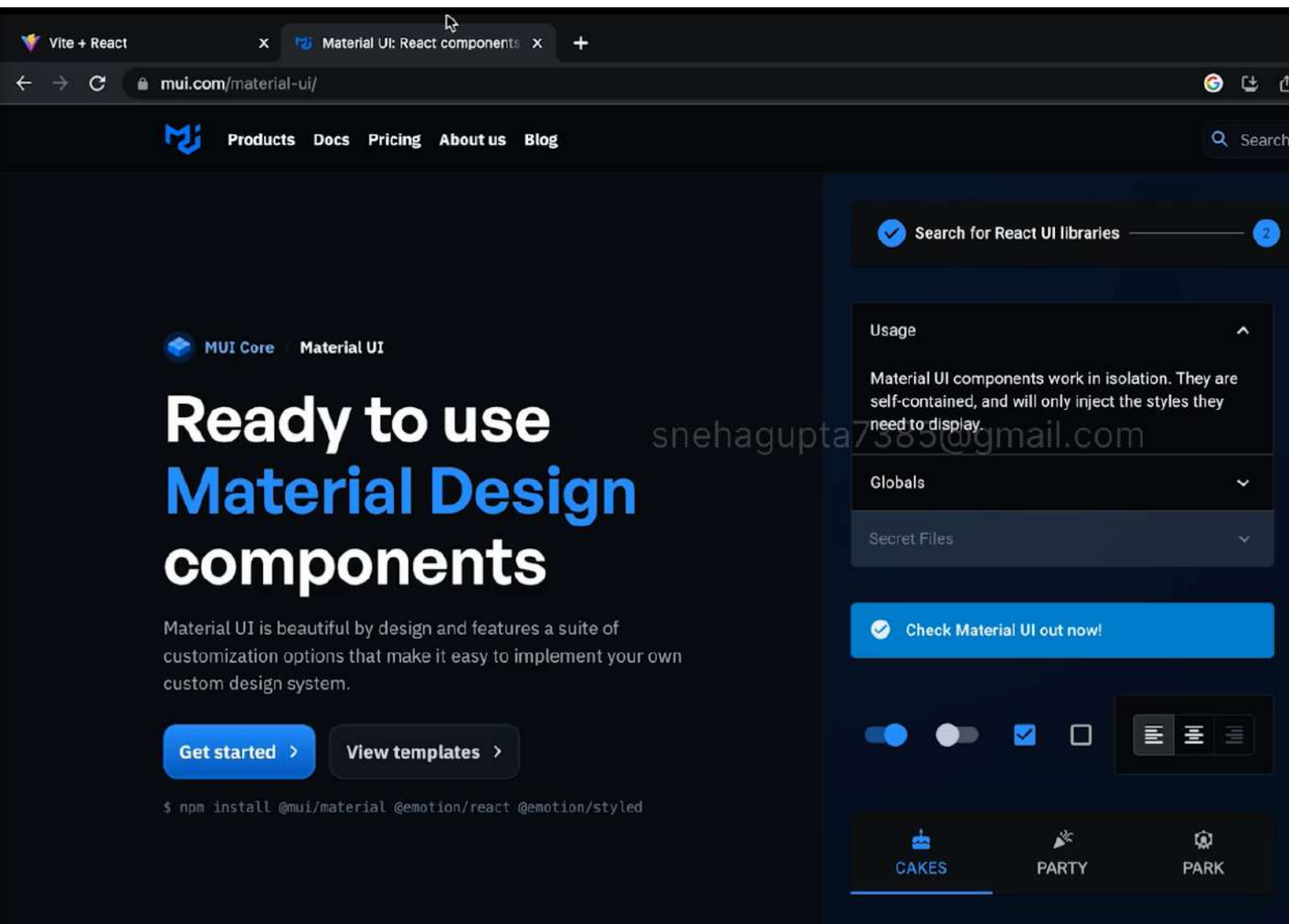


Material UI

Library of React UI components



Vite + React

Installation - Material UI

mui.com/material-ui/getting-started/installation/

MUI CORE

Material UI v5.14.17

Getting started

Overview

Installation

Usage

Example projects

Templates

Learn

Design resources

FAQs

Supported components

Supported platforms

Support

Components

Component API

Customization

How-to guides

Experimental APIs

Discover more

Migration

Templates

Installation

Install Material UI, the world's most popular React UI framework.

Default installation

Run one of the following commands to add Material UI to your project:

```
npm yarn pnpm
```

```
npm install @mui/material @emotion/react @emotion/styled
```

With styled-components

Material UI uses [Emotion](#) as its default styling engine. If you want to use [styled-components](#) instead, run one of the following commands:

```
npm yarn pnpm
```

```
npm install @mui/material @mui/styled-engine-sc styled-components
```

Next, follow the [styled-components how-to guide](#) to properly configure your bundler to support `@mui/styled-engine-sc`.

Default installation

Run one of the following commands to add Material UI to your project:

npm yarn pnpm

```
npm install @mui/material @emotion/react @emotion/styled
```

Peer dependencies

Please note that [react](#) and [react-dom](#) are peer dependencies, meaning you should ensure they are installed before installing Material UI.

```
"peerDependencies": {  
  "react": "^17.0.0 || ^18.0.0",  
  "react-dom": "^17.0.0 || ^18.0.0"  
},
```

With styled-components

Material UI uses [Emotion](#) as its default styling engine. If you want to use [styled-components](#) instead, run one of the following commands:

npm yarn pnpm

```
npm install @mui/material @mui/styled-engine-sc styled-components
```

Roboto font



Material UI uses the [Roboto](#) font by default. Add it to your project via Fontsource, or with the Google Fonts CDN.

npm yarn pnpm


```
npm install @fontsource/roboto
```



Then you can import it in your entry point like this:

```
import '@fontsource/roboto/300.css';  
import '@fontsource/roboto/400.css';  
import '@fontsource/roboto/500.css';  
import '@fontsource/roboto/700.css';
```



-  Fontsource can be configured to load specific subsets, weights and styles. Material UI's default typography configuration relies only on the 300, 400, 500, and 700 font weights.

Icons



To use the [font Icon component](#) or the prebuilt SVG Material Icons (such as those found in the [icon demos](#)), you must first install the [Material Icons](#) font. You can do so with npm, or with the Google Web Fonts CDN.

npm yarn pnpm

```
npm install @mui/icons-material
```



Google Web Fonts



To install the Material Icons font in your project using the Google Web Fonts CDN, add the following code snippet inside your project's `<head />` tag:

To use the font `Icon` component, you must first add the [Material Icons](#) font. Here are [some instructions](#) on how to do so. For instance, via Google Web Fonts:

```
<link
  rel="stylesheet"
  href="https://fonts.googleapis.com/icon?family=Material+Icons"
/>
```



Basic button



The `Button` comes with three variants: text (default), contained, and outlined.

TEXT

CONTAINED

OUTLINED

JS

TS

Collapse code



```
import * as React from 'react';
import Stack from '@mui/material/Stack';
import Button from '@mui/material/Button';

export default function BasicButtons() {
  return (
    <Stack spacing={2} direction="row">
      <Button variant="text">Text</Button>
      <Button variant="contained">Contained</Button>
      <Button variant="outlined">Outlined</Button>
    </Stack>
  );
}
```

Text button



[Text buttons](#) are typically used for less-pronounced actions, including those located: in dialogs, in cards. In cards, text buttons help maintain an emphasis on card content.

PRIMARY DISABLED LINK

JS

TS

Collapse code



```
import * as React from 'react';
import Button from '@mui/material/Button';
import Stack from '@mui/material/Stack';

export default function TextButtons() {
  return (
    <Stack direction="row" spacing={2}>
      <Button>Primary</Button>
      <Button disabled>Disabled</Button>
      <Button href="#text-buttons">Link</Button>
    </Stack>
  );
}
```


Color



SECONDARY

SUCCESS

ERROR

JS

TS

Collapse code



```
import * as React from 'react';
import Stack from '@mui/material/Stack';
import Button from '@mui/material/Button';

export default function ColorButtons() {
  return (
    <Stack direction="row" spacing={2}>
      <Button color="secondary">Secondary</Button>
      <Button variant="contained" color="success">
        Success
      </Button>
      <Button variant="outlined" color="error">
        Error
      </Button>
    </Stack>
  );
}
```

Sizes

For larger or smaller buttons, use the `size` prop.

SMALL

MEDIUM

LARGE

SMALL

MEDIUM

LARGE

SMALL

MEDIUM

LARGE

Show code



Buttons with icons and label

Sometimes you might want to have icons for certain buttons to enhance the UX of the application as we recognize logos more easily than plain text. For example, if you have a delete button you can label it with a dustbin icon.



JS

TS

Collapse code



```
import * as React from 'react';
import Button from '@mui/material/Button';
import DeleteIcon from '@mui/icons-material/Delete';
import SendIcon from '@mui/icons-material/Send';
import Stack from '@mui/material/Stack';

export default function IconLabelButtons() {
  return (
    <Stack direction="row" spacing={2}>
      <Button variant="outlined" startIcon={<DeleteIcon />}>
        Delete
      </Button>
      <Button variant="contained" endIcon={<SendIcon />}>
        Send
      </Button>
    </Stack>
  );
}
```

Sizes

For larger or smaller icon buttons, use the `size` prop.



JS

TS

Collapse code



```
import * as React from 'react';
import Stack from '@mui/material/Stack';
import IconButton from '@mui/material/IconButton';
import DeleteIcon from '@mui/icons-material/Delete';

export default function IconButtonSizes() {
  return (
    <Stack direction="row" alignItems="center" spacing={1}>
      <IconButton aria-label="delete" size="small">
        <DeleteIcon fontSize="inherit" />
      </IconButton>
      <IconButton aria-label="delete" size="small">
        <DeleteIcon fontSize="small" />
      </IconButton>
      <IconButton aria-label="delete" size="large">
        <DeleteIcon />
      </IconButton>
      <IconButton aria-label="delete" size="large">
        <DeleteIcon fontSize="inherit" />
      </IconButton>
    </Stack>
  );
}
```

File upload



To create a file upload button, turn the button into a label using `component="label"` and then create a visually-hidden input with type `file`.

 UPLOAD FILE[Expand code](#)

```
<Button
  component="label"
  role={undefined}
  variant="contained"
  tabIndex={-1}
  startIcon={<CloudUploadIcon />}
>
  Upload file
  <VisuallyHiddenInput type="file" />
</Button>
```

Customization



Here are some examples of customizing the component. You can learn more about this in the [overrides documentation page](#).

CUSTOM CSS

Bootstrap

Show code



Building a Weather Widget

using React & Material UI

Building a Weather Widget

using React & Material UI

form (city)

Delhi, Pune, Mumbai

Search



temp

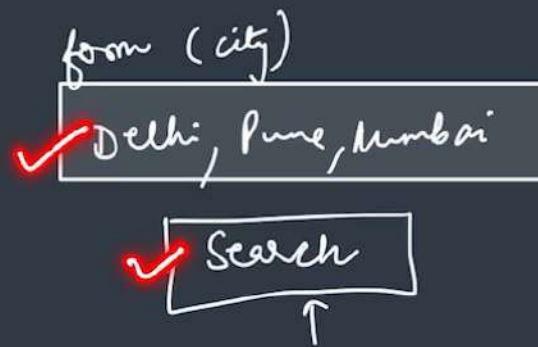
humidity

min-
max

feels like

Building a Weather Widget

using React & Material UI



API
+
Data
print



```

1  import {useState} from "react"
2  import Button from '@mui/material/Button';
3  import TextField from '@mui/material/TextField';
4  import SearchIcon from '@mui/icons-material/Search';
5
6  Complexity is 13 You must be kidding
7  export default function SearchBox(){
8      let [cityname,setCityname]=useState("");
9      let API_URL="https://api.openweathermap.org/data/2.5/weather";
10     let API_Key="07b15329fb5273208bc0a48a9f105e30";
11
12     let getWeather=async ()=>{
13         let response = await fetch(`${API_URL}?q=${cityname}&appid=${API_Key}&units=metric`);
14         let JSONresponse=await response.json();
15         console.log(JSONresponse);
16         let result={
17             country:JSONresponse.sys.country,
18             city:cityname,
19             lat:JSONresponse.coord.lat,
20             lon:JSONresponse.coord.lon,
21             humidity:JSONresponse.main.humidity,
22             temp:JSONresponse.main.temp,
23             temp_max:JSONresponse.main.temp_max,
24             temp_min:JSONresponse.main.temp_min,
25             feels_like:JSONresponse.main.feels_like,
26             weather:JSONresponse.weather[0].main,
27             wind:JSONresponse.wind.speed,
28             pressure:JSONresponse.main.pressure,
29         }
30         console.log(result);
31     }
32
33     let handlecity=(event)=>{
34         setCityname(event.target.value);
35     };
36     let handlesubmit=(evt)=>{
37         evt.preventDefault();
38         console.log(cityname);
39         getWeather(cityname);
40         setCityname("");
41     };
42     return(
43         <div>
44             <h4>Search for the Weather</h4>
45             <form onSubmit={handlesubmit}>
46                 <TextField required id="cityname" label="Search city" variant="outlined" value={cityname} onChange={handlecity}/>
47                 <br /><br />
48                 <Button variant="contained" type="submit" startIcon={<SearchIcon/>}>Search</Button>
49             </form>
50         </div>
51     );
52 }

```

Search for the Weather

Search city *

Switzerland

SEARCH

Switzerland

SearchBox.jsx:37

SearchBox.jsx:14

```
{coord: {...}, weather: Array(1), base: 'stations', main: {...}, visibility: 10000}
0, ...}
  base: "stations"
  clouds: {all: 74}
  cod: 200
  coord: {lon: 8.0143, lat: 47.0002}
  dt: 1720160075
  id: 2658434
  main: {temp: 13.55, feels_like: 13.15, temp_min: 12.15, temp_max: 14.7, pressure: 1016, humidity: 84, wind_speed: 0.44, wind_deg: 262, clouds: 74, visibility: 10000}
  name: "Switzerland"
  sys: {type: 2, id: 2034458, country: 'CH', sunrise: 1720150774, sunset: 1720150774, timezone: 7200}
  visibility: 10000
  weather: [{...}]
  wind: {speed: 0.44, deg: 262, gust: 0.87}
  [[Prototype]]: Object
```

SearchBox.jsx:29

```
{country: 'CH', city: 'Switzerland', lat: 47.0002, lon: 8.0143, humidity: 84, pressure: 1016, temp: 13.55, temp_max: 14.7, temp_min: 12.15, weather: 'Clouds', wind: 0.44}
4, ...}
  city: "Switzerland"
  country: "CH"
  feels_like: 13.15
  humidity: 84
  lat: 47.0002
  lon: 8.0143
  pressure: 1016
  temp: 13.55
  temp_max: 14.7
  temp_min: 12.15
  weather: "Clouds"
  wind: 0.44
  [[Prototype]]: Object
```