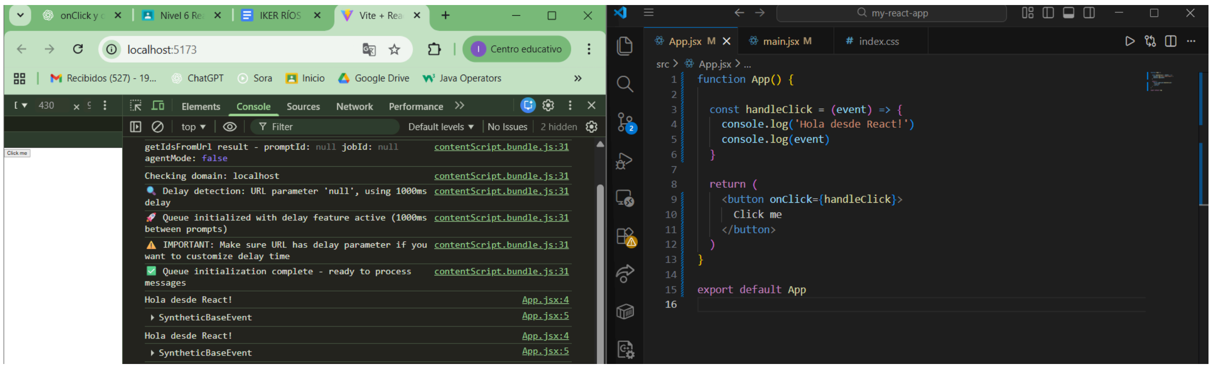
Nivel 6 React - 21/01/2026 - Iker Ríos Navarro

ÍNDICE

Parte A: onClick y el objeto event:

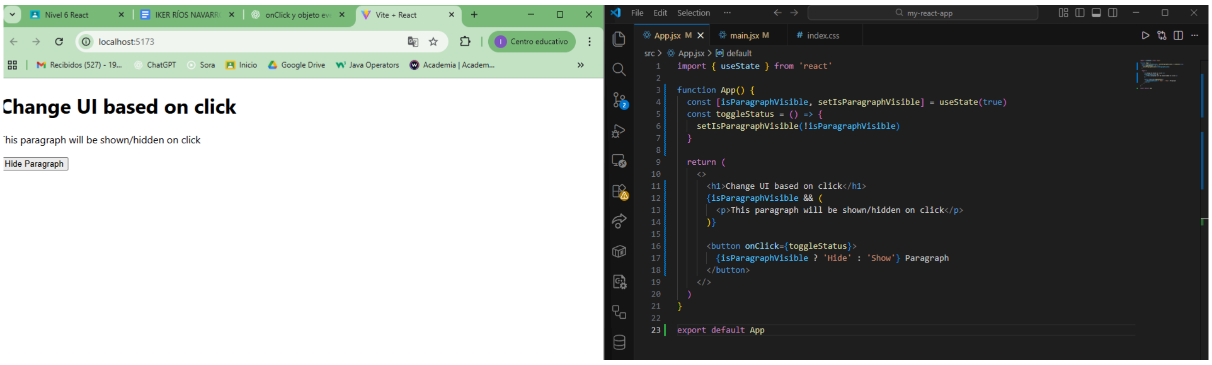


```
getIdsFromUrl result - promptId: null jobId: null         contentScript.bundle.js:31
agentMode: false
Checking domain: localhost                                 contentScript.bundle.js:31
🔍 Delay detection: URL parameter 'null', using 1000ms     contentScript.bundle.js:31
delay
🚀 Queue initialized with delay feature active (1000ms     contentScript.bundle.js:31
between prompts)
⚠️ IMPORTANT: Make sure URL has delay parameter if you     contentScript.bundle.js:31
want to customize delay time
✅ Queue initialization complete - ready to process        contentScript.bundle.js:31
messages
Hola desde React!                                                        App.jsx:4
 ▸ SyntheticBaseEvent                                                    App.jsx:5
Hola desde React!                                                        App.jsx:4
 ▸ SyntheticBaseEvent                                                    App.jsx:5
```

```jsx
function App() {

  const handleClick = (event) => {
    console.log('Hola desde React!')
    console.log(event)
  }

  return (
    <button onClick={handleClick}>
      Click me
    </button>
  )
}

export default App
```

(Captura del código por elaboración propia)

Parte B: Cambiar la UI con estado + evento:

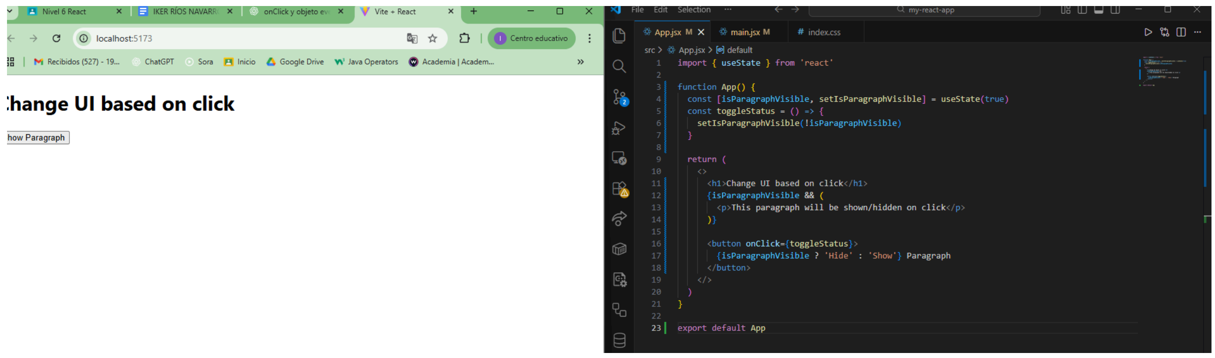


Change UI based on click

his paragraph will be shown/hidden on click

Hide Paragraph

```jsx
import { useState } from 'react'

function App() {
  const [isParagraphVisible, setIsParagraphVisible] = useState(true)
  const toggleStatus = () => {
    setIsParagraphVisible(!isParagraphVisible)
  }

  return (
    <>
      <h1>Change UI based on click</h1>
      {isParagraphVisible && (
        <p>This paragraph will be shown/hidden on click</p>
      )}

      <button onClick={toggleStatus}>
        {isParagraphVisible ? 'Hide' : 'Show'} Paragraph
      </button>
    </>
  )
}

export default App
```



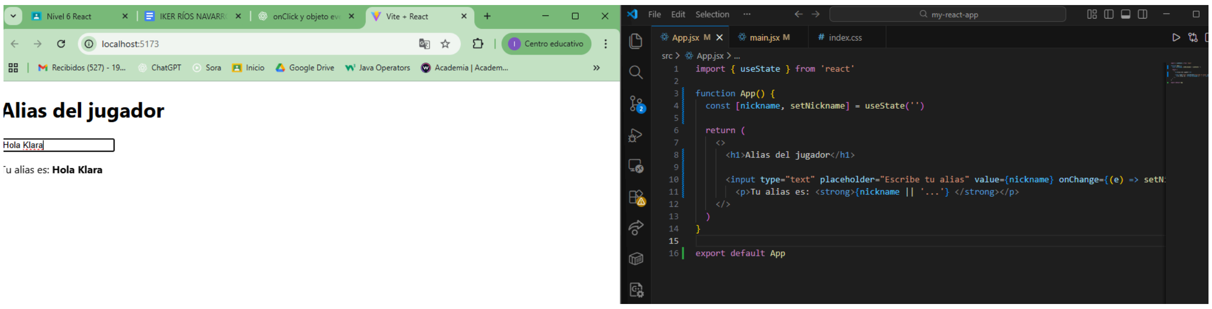
hange UI based on click

how Paragraph

```jsx
import { useState } from 'react'

function App() {
  const [isParagraphVisible, setIsParagraphVisible] = useState(true)
  const toggleStatus = () => {
    setIsParagraphVisible(!isParagraphVisible)
  }

  return (
    <>
      <h1>Change UI based on click</h1>
      {isParagraphVisible && (
        <p>This paragraph will be shown/hidden on click</p>
      )}

      <button onClick={toggleStatus}>
        {isParagraphVisible ? 'Hide' : 'Show'} Paragraph
      </button>
    </>
  )
}

export default App
```

(Captura del código por elaboración propia)

Parte C: onChange (entrada controlada):



Alias del jugador

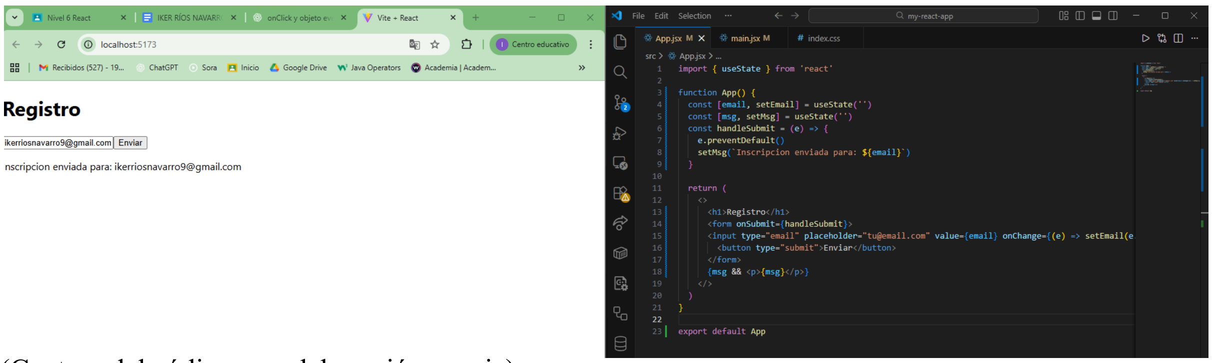
Hola Klara

u alias es: **Hola Klara**

```jsx
import { useState } from 'react'

function App() {
  const [nickname, setNickname] = useState('')

  return (
    <>
      <h1>Alias del jugador</h1>

      <input type="text" placeholder="Escribe tu alias" value={nickname} onChange={(e) => setN
        <p>Tu alias es: <strong>{nickname || '...'} </strong></p>
    </>
  )
}

export default App
```

(Captura del código por elaboración propia)

Parte D: onSubmit (formulario):



Registro

```
ikerriosnavarro9@gmail.com  Enviar

nscripcion enviada para: ikerriosnavarro9@gmail.com
```

```jsx
import { useState } from 'react'

function App() {
  const [email, setEmail] = useState('')
  const [msg, setMsg] = useState('')
  const handleSubmit = (e) => {
    e.preventDefault()
    setMsg(`Inscripcion enviada para: ${email}`)
  }

  return (
    <>
      <h1>Registro</h1>
      <form onSubmit={handleSubmit}>
      <input type="email" placeholder="tu@email.com" value={email} onChange={(e) => setEmail(e
        <button type="submit">Enviar</button>
      </form>
      {msg && <p>{msg}</p>}
    </>
  )
}

export default App
```

(Captura del código por elaboración propia)

Preguntas:

1. ¿Qué diferencia hay entre onClick y onSubmit?

onClick actúa sobre elementos individuales, mientras que onSubmit gestiona el envío completo de un formulario.

2. ¿Por qué usamos e.preventDefault() en un formulario?

preventDefault() evita el comportamiento por defecto del navegador y permite controlar el formulario desde React.

3. ¿Qué es una "entrada controlada" y por qué usamos value + onChange?

Usamos value + onChange para que React tenga el control total del input.

4. En tu mini-reto, que estado(s) manejas y que evento(s) los actualizan?

Los estados se actualizan mediante eventos onClick asociados a botones que modifican el estado con setState.