TypeScript + React Cheat Sheet

React Functional Component

```
type Props = {
  title: string
  count?: number
}

const MyComponent: React.FC<Props> = ({ title, count = 0 }) => {
  return <h1>{title}: {count}</h1>
}
```

useState

```
const [name, setName] = useState<string>('John')
const [count, setCount] = useState<number>(0)
const [data, setData] = useState<{ id: number; value: string } | null>(null)
```

useEffect

```
useEffect(() => {
  console.log("Component mounted")
}, [])
```

Event Handlers

```
const handleClick = (e: React.MouseEvent<HTMLButtonElement>) => {}
const handleChange = (e: React.ChangeEvent<HTMLInputElement>) => {}
```

Refs

```
const inputRef = useRef<HTMLInputElement>(null)
<input ref={inputRef} />
```

Forms

```
type FormData = {
  email: string
  password: string
}

const [form, setForm] = useState<FormData>({ email: '', password: '' })
```

Children Prop

```
type Props = {
  children: React.ReactNode
}

const Container: React.FC<Props> = ({ children }) => {
  return <div>{children}</div>
}
```

Union Props

```
type ButtonProps =
  | { variant: 'primary'; onClick: () => void }
  | { variant: 'link'; href: string }

const Button: React.FC<ButtonProps> = (props) => {
  if (props.variant === 'primary') {
    return <button onClick={props.onClick}>Click me</button>
  } else {
    return <a href={props.href}>Go to link</a>
  }
}
```

Typing API Data

```
type User = {
  id: number
  name: string
}

const [user, setUser] = useState<User | null>(null)
```

Typing Context API

```
type Theme = 'light' | 'dark'
const ThemeContext = React.createContext<Theme>('light')
const useTheme = () => useContext(ThemeContext)
```

Typing useReducer

```
type State = { count: number }
type Action = { type: 'increment' | 'decrement' }

const reducer = (state: State, action: Action): State => {
  switch (action.type) {
    case 'increment': return { count: state.count + 1 }
    case 'decrement': return { count: state.count - 1 }
  }
}

const [state, dispatch] = useReducer(reducer, { count: 0 })
```