Example: Restaurant App with Shimmer UI (Loading State)

Goal:

> On page load → show Shimmer → fetch restaurant data (mock API) → update UI with real data.

Folder Structure:

```
/src
├── App.js
├── components/
│   ├── Body.js
│   ├── RestaurantCard.js
│   └── Shimmer.js
```

App.js – Root File

```jsx
import React from "react";
import Body from "./components/Body";

const App = () => {
  return (
    <div>
      <h1>🍽️ Restaurant Explorer</h1>
      <Body />
    </div>
  );
};

export default App;
```

components/Shimmer.js – Loader UI

```jsx
import React from "react";

const Shimmer = () => {
  return (
    <div className="shimmer-container">
      {[...Array(5)].map((_, index) => (
        <div className="shimmer-card" key={index}></div>
```

```
      ))}
    </div>
  );
};

export default Shimmer;
```

🧠 **Explanation**:

- Shows 5 shimmer cards (dummy boxes).
- Mimics layout while data is being fetched.

📁 components/RestaurantCard.js – Display Each Restaurant

```
import React from "react";

const RestaurantCard = ({ name, cuisine, rating }) => {
  return (
    <div className="restaurant-card">
      <h3>{name}</h3>
      <p>{cuisine}</p>
      <p>⭐ {rating}</p>
    </div>
  );
};

export default RestaurantCard;
```

📁 components/Body.js – Main Logic Component

```
import React, { useState, useEffect } from "react";
import Shimmer from "./Shimmer";
import RestaurantCard from "./RestaurantCard";

const Body = () => {
  const [restaurants, setRestaurants] = useState(null); // initially null

  // 🚀 useEffect triggers only once after initial render
  useEffect(() => {
    fetchRestaurantData();
  }, []);

  const fetchRestaurantData = async () => {
    // ⏳ Simulate API delay
    const data = await new Promise((resolve) =>
      setTimeout(() => {
        resolve([
```

```
            { id: 1, name: "Biryani House", cuisine: "Indian", rating: 4.3 },
            { id: 2, name: "Pizza Palace", cuisine: "Italian", rating: 4.6 },
            { id: 3, name: "Sushi Station", cuisine: "Japanese", rating: 4.7 },
          ]);
        }, 2000) // 2 sec delay
      );

      setRestaurants(data); // ☑️ update UI with actual data
    };

    // 🧠 Early return pattern
    if (!restaurants) {
      return <Shimmer />;
    }

    return (
      <div className="restaurant-list">
        {restaurants.map((rest) => (
          <RestaurantCard key={rest.id} {...rest} />
        ))}
      </div>
    );
  };

  export default Body;
```

🧠 Workflow Breakdown

Step	What Happens
🖥️ Page Loads	React renders `App` and `Body` component
📦 Initial State	`restaurants = null` ➤ triggers early return
🔄 Render	`<Shimmer />` is shown instantly
🔁 `useEffect` Fires	API call (simulated by `setTimeout`) begins
☑️ After 2 sec	`setRestaurants(data)` updates state
🔁 Re-render	React re-renders `Body` and shows `<RestaurantCard>`s

🎨 Basic CSS (optional for shimmer)

```
.shimmer-container {
  display: flex;
  gap: 20px;
}

.shimmer-card {
  width: 200px;
```

```
  height: 120px;
  background: linear-gradient(90deg, #eee, #ddd, #eee);
  animation: shimmer 1.5s infinite;
  border-radius: 8px;
}

@keyframes shimmer {
  0% {
    background-position: -200px;
  }
  100% {
    background-position: 200px;
  }
}

.restaurant-card {
  border: 1px solid #ccc;
  border-radius: 8px;
  padding: 12px;
  margin: 10px;
  background: #f9f9f9;
}
```

🧠 Why This Pattern is Powerful:

- ☑ Prevents crashes on undefined data (`null`)
- ☑ Shows immediate visual feedback (`Shimmer`)
- ☑ Improves UX by giving the user a sense that “something is happening”
- ☑ Keeps JSX clean via early return or optional chaining

💎 Final Takeaways

Concept	Explanation
`useState(null)`	Signals data is not fetched yet
`useEffect()`	Used to make API calls after initial render
`Early Return`	Prevents rendering errors when data is not ready
`Shimmer UI`	Placeholder loading layout
`Mock API`	Simulated using `setTimeout()` for demo
