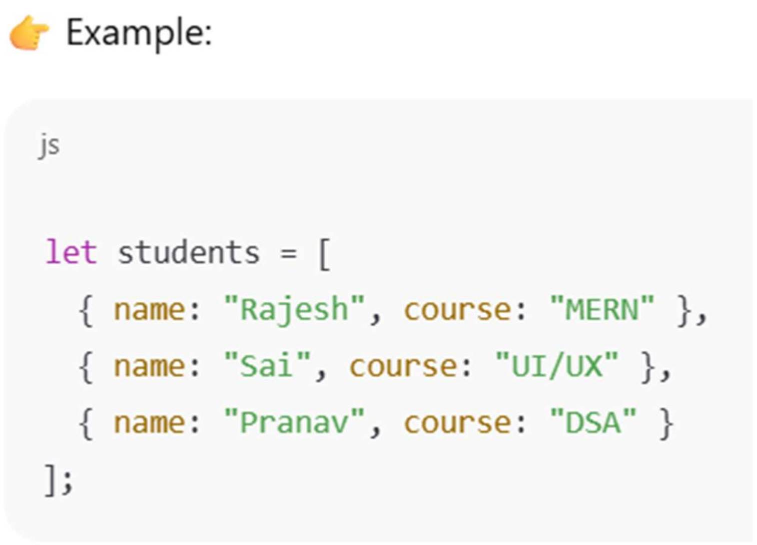
🗒 Session Notes – 24-09-2025

1. Looping Through an Array of Objects

In real-world projects (like databases), we often deal with arrays of objects.



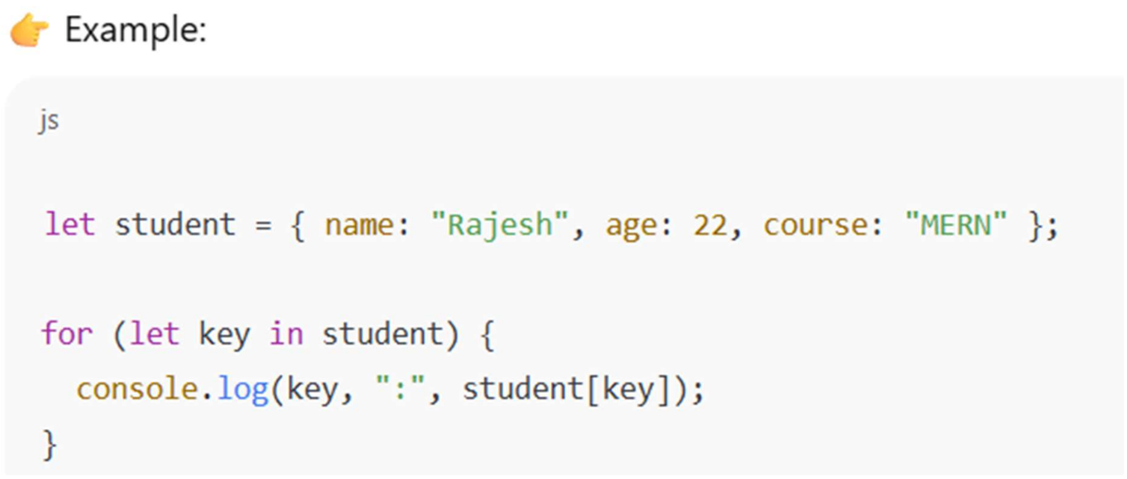
👉 Example:

js

```js
let students = [
  { name: "Rajesh", course: "MERN" },
  { name: "Sai", course: "UI/UX" },
  { name: "Pranav", course: "DSA" }
];
```

for...in loop

- Iterates over the **keys/indexes** of an object/array.
- Best used for **objects**, not arrays (but still works with indexes).



👉 Example:

js

```js
let student = { name: "Rajesh", age: 22, course: "MERN" };

for (let key in student) {
  console.log(key, ":", student[key]);
}
```

}

✅ Output:

yaml

```yaml
name : Rajesh
age : 22
course : MERN
```

---

for...of loop

- Iterates over the **values** in an iterable (like arrays, strings).
- Very useful for looping arrays of objects.

👉 Example:

js

```js
for (let s of students) {
  console.log(s.name, "-", s.course);
}
```

✅ Output:

nginx

```nginx
Rajesh - MERN
Sai - UI/UX
Pranav - DSA
```

📍 **Key Difference**:

- `for…in` → keys/indexes
- `for…of` → values

---

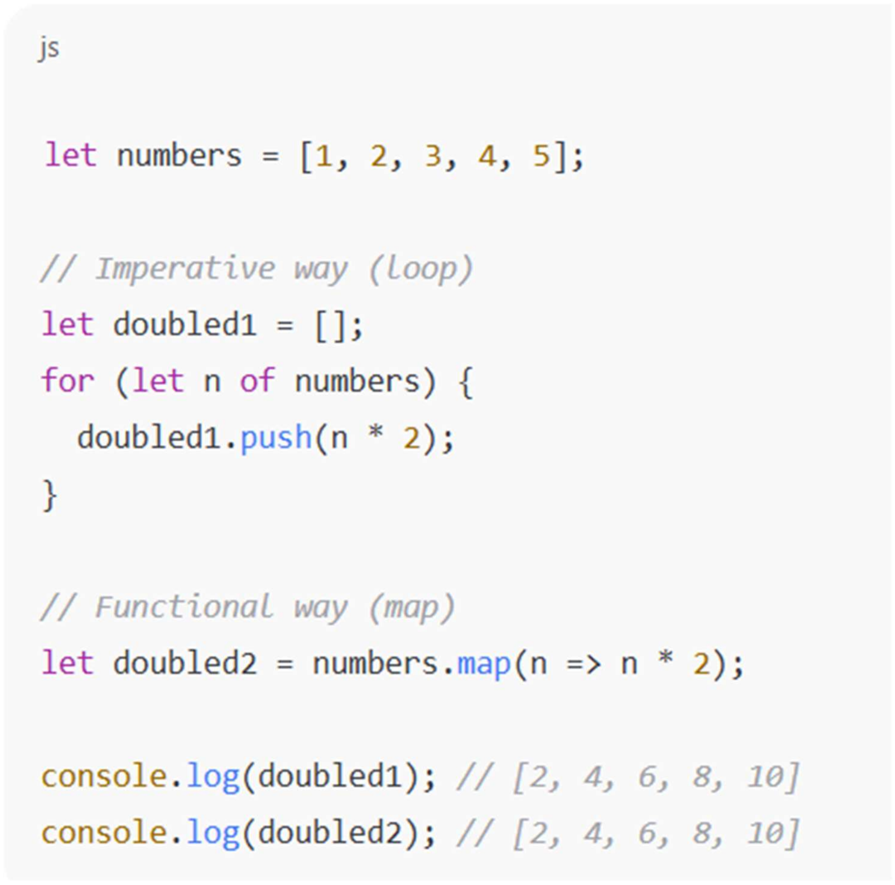
2. Introduction to Functional Programming (FP)

Functional Programming = **writing code using pure functions, immutability, and reusability**.

Core Principles of FP in JavaScript

1. **Pure Functions** – Same input = same output, no side effects.
2. **Immutability** – Avoid changing original data, instead return new values.
3. **Higher-Order Functions** – Functions that take other functions as arguments or return functions (`map`, `filter`, `reduce`).
4. **Declarative Style** – Focus on *what* to do, not *how* to do it.

👉 Example:

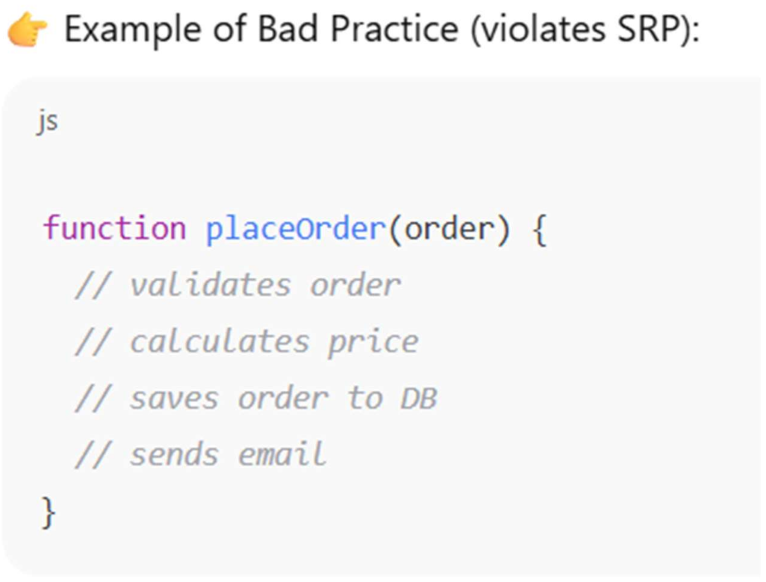


```js
let numbers = [1, 2, 3, 4, 5];

// Imperative way (loop)
let doubled1 = [];
for (let n of numbers) {
  doubled1.push(n * 2);
}

// Functional way (map)
let doubled2 = numbers.map(n => n * 2);

console.log(doubled1); // [2, 4, 6, 8, 10]
console.log(doubled2); // [2, 4, 6, 8, 10]
```

---

3. Single Responsibility Principle (SRP)

- From **SOLID principles** in software engineering.
- **Definition:** A function/class/module should have **only one reason to change** → it should do **one job only**.



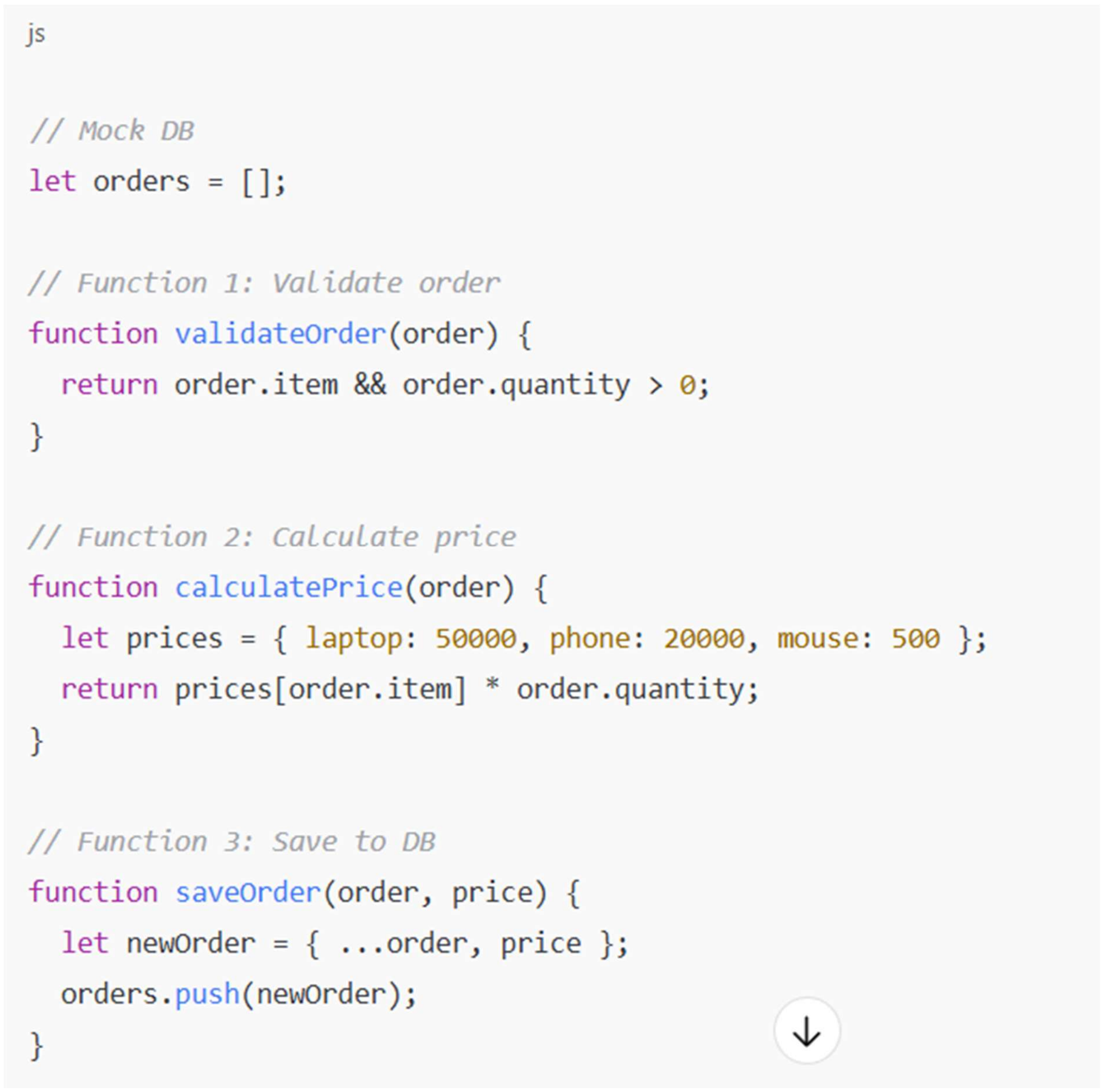
👉 Example of Bad Practice (violates SRP):

```js
function placeOrder(order) {
  // validates order
  // calculates price
  // saves order to DB
  // sends email
}
```

⚠ Too many responsibilities in one function.

Nano Project: Ecommerce – Placing an Order

☞ Applying **SRP** + **Functional Programming**:



js

```js
// Mock DB
let orders = [];

// Function 1: Validate order
function validateOrder(order) {
  return order.item && order.quantity > 0;
}

// Function 2: Calculate price
function calculatePrice(order) {
  let prices = { laptop: 50000, phone: 20000, mouse: 500 };
  return prices[order.item] * order.quantity;
}

// Function 3: Save to DB
function saveOrder(order, price) {
  let newOrder = { ...order, price };
  orders.push(newOrder);
}
```

```
// Function 4: Place Order (SRP: just orchestrates)
function placeOrder(order) {
  if (!validateOrder(order)) return "Invalid order!";
  let price = calculatePrice(order);
  saveOrder(order, price);
  return "Order placed successfully!";
}

// Test
console.log(placeOrder({ item: "laptop", quantity: 2 }));
console.log(orders);
```

✅ Output:

```css
Order placed successfully!
[ { item: 'laptop', quantity: 2, price: 100000 } ]
```

✓ Conclusion

- **for…in** → iterate over keys/properties.
- **for…of** → iterate over values (useful for arrays).
- Functional programming → pure functions, immutability, higher-order functions.
- **SRP** ensures each function/class does **only one job**, improving maintainability.
- Applied SRP in an **Ecommerce Order Simulation** nano-project.

🎯 Interview Prep – Loops, Functional Programming & SRP

1. Loops with Arrays & Objects

1. What is the difference between `for…in` and `for…of` loops in JavaScript?
2. When should you use `for…in` instead of `for…of`?
3. Write code to iterate over an **object's properties** using `for…in`.
4. Write code to iterate over an **array of student objects** using `for…of`.
5. Can you use `for…in` on arrays? What is the drawback?

2. Functional Programming (FP)

6. What is **functional programming**? How is it different from imperative programming?
7. What is a **pure function**? Give an example in JS.
8. Explain **immutability** with a JavaScript example.
9. What are **higher-order functions**? Name a few built-in HOFs in JavaScript.
10. Rewrite a loop-based array operation (`double numbers`) using **map()** (functional way).

3. Single Responsibility Principle (SRP)

11. What is the **Single Responsibility Principle (SRP)**?
12. Why is SRP important in real-world projects?
13. Give an example of a function that violates SRP.
14. Refactor the same function into multiple functions following SRP.
15. How would you explain SRP to a non-technical person?

4. Practical Coding Questions

16. Write a program that takes an array of student objects and prints only their names using `for…of`.
17. Write a function `getKeys(obj)` that returns all keys of an object using `for…in`.
18. Write a program to **filter orders greater than ₹50,000** from an array of orders using `filter()` (FP).

19. Implement a small **Ecommerce "placeOrder" simulation** where you split responsibilities (validation, price calculation, saving).
20. Refactor the above program to follow **SRP** – one function per responsibility.