**Callback Hell** ka scene ekdum **desi style** me:

**1. Concept – Callback Hell kya hai?**

👉 Callback Hell tab hota hai jab tum ek ke andar ek callback chipkate jaate ho.

* Matlab ek kaam khatam hote hi dusra
* Dusra khatam hote hi teesra
* Aur aise hi chain ban jaati hai...

Result → code ka shape ban jaata hai **“pyramid of doom”** ⛺

**2. Desi Example**

Socho tum Pizza order karte ho 🍕:

1. Pehle **base banani hai**
2. Fir uspe **sauce lagana hai**
3. Fir **cheese dalni hai**
4. Fir oven me dalna hai
5. Fir serve karna hai

Agar har step async ho aur callback me likha ho, toh code aisa ho jaata hai:

makeBase(() => {

addSauce(() => {

addCheese(() => {

bake(() => {

serve(() => {

console.log("Pizza ready ho gaya Bhai! 🍕");

});

});

});

});

});

👉 Dekha? Yeh hi hai **callback hell**. Code ganda lagta hai, samajhna mushkil hota hai, aur maintain karna aur bhi mushkil.

**3. Coding Example (Callback Hell)**

console.log("Kaam shuru...");

setTimeout(() => {

console.log("Step 1: Paani garam ho gaya 💧");

setTimeout(() => {

console.log("Step 2: Chai patti daal di ☕");

setTimeout(() => {

console.log("Step 3: Doodh daal diya 🥛");

setTimeout(() => {

console.log("Step 4: Chini daal di 🍬");

console.log("Chai taiyaar ho gayi bhai! ✅");

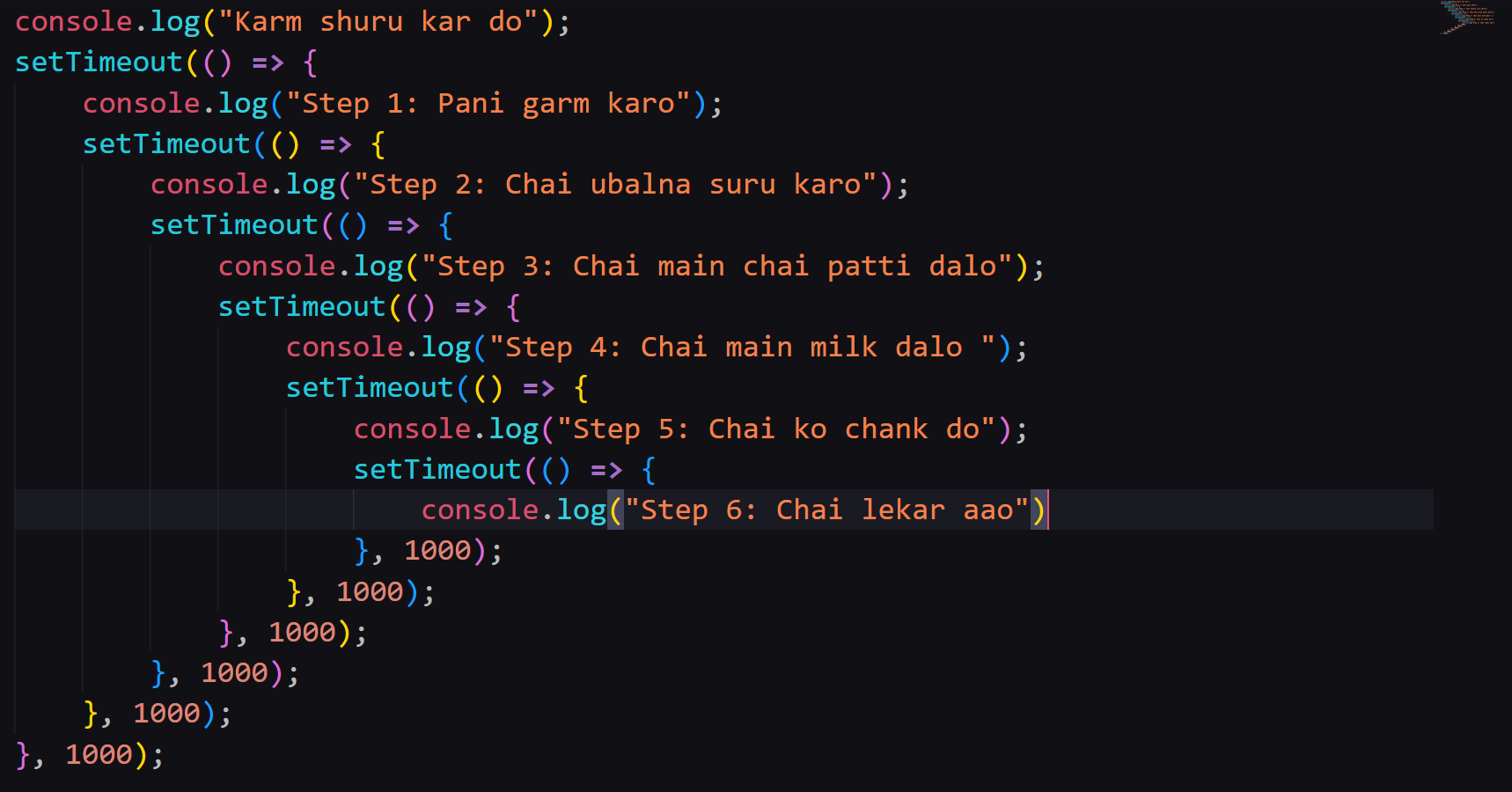
}, 1000);

}, 1000);

}, 1000);

}, 1000);

👉 Output line-by-line aayega, lekin code dekhke lag raha hai **ladder chadh rahe ho** 🪜.



**4. Problem kahan hai?**

* Code **nested** hota jaata hai
* Read karna mushkil
* Maintain karna mushkil
* Error handling kaafi confusing ho jaata hai

**5. Desi Summary**

* **Callback Hell** = ek ke andar ek callback → code pyramid ban jaata hai
* Example = Pizza order / Chai banane ka process
* Solution = **Promises** aur **Async/Await**