**Error Handling Strategies in JavaScript**

**What is an Error?**

An **error** is a **mistake in code** that stops your program or causes it to behave wrongly.

**Example in real life:**  
If you try to divide something by zero in math — you get an error.  
In JavaScript, it’s the same — the browser will stop running your code if there’s an error.

**1. Error Types in JavaScript**

**a) Syntax Error**

Mistake in **code writing rules** (grammar of JavaScript)

// Missing closing bracket

console.log("Hello" // ❌ Syntax Error

**b) Runtime Error**

Code is written correctly, but fails **when it runs**

let num = 10;

console.log(num.toUpperCase()); // ❌ num is a number, not a string → Runtime error

**c) Logical Error**

Code runs, but gives **wrong output** — logic is wrong.

let a = 5, b = 10;

console.log(a + b); // ✅ 15

// But if you mistakenly write:

console.log(a \* b); // ❌ 50 instead of 15 (logical mistake)

**2. Handling Errors Effectively**

**Using try...catch block**

Helps you **catch errors safely** and **keep the program running**.

**Basic Syntax:**

try {

// risky code

} catch (error) {

// handle error here

}

**Example:**

try {

let x = y + 10; // y is not defined → error

} catch (err) {

console.log("Something went wrong:", err.message);

}

Even if error happens, the browser doesn’t crash — because we handled it safely!

**Creating Custom Error with throw**

You can **manually throw your own error** using throw.

**Example:**

let age = -5;

try {

if (age < 0) {

throw new Error("Age cannot be negative!"); // custom error

}

console.log("Age is:", age);

} catch (err) {

console.log("Error:", err.message);

}