🧠 Handling HTTP Status Codes — ENGINEER MODE (HINGLISH)

Core Idea (1 line)

`requests.get()` **sirf data nahi deta — decision lene ka signal deta hai.**

Ab hum code ko **blind nahi**, **aware** banayenge.

🔍 Step 1: Response ke 3 important signals

Jab bhi request bhejo, **yeh teen cheezein dekho**:

```
response.status_code
response.text
response.headers
```

Aaj sirf **status_code** pe focus.

🟢 Rule-Based Thinking (SYSTEM DESIGN)

Socho jaise traffic signal 🚦:

☑ 200 → Green

- Kaam karo
- HTML parse karo

⚠ 404 → Dead end

- Page exist nahi
- Loop break ya skip

🚫 403 → Entry banned

- Aage mat badho
- Scraper ko **respectfully stop** karo

🐌 429 → Slow down

- Thoda wait
- Retry later

🔥 5xx → Server problem

- Tumhari galti nahi
- Retry OR skip

🧩 Step 2: Minimum Safe `fetch_html()` (ENGINEER VERSION)

❌ Tumhara current version (problematic)

```
def fetch_html(self, url):
    response = requests.get(url, headers=self.headers)
    return response.text
```

❓ Isme kya galat hai?

- 403 bhi aaye → tum HTML parse karoge
- 404 bhi aaye → tum parse karoge
- Internet down → crash

☑ Step 3: Proper Handling (READ CAREFULLY)

```
import requests

def fetch_html(self, url):
    try:
        response = requests.get(url, headers=self.headers,
timeout=10)

        if response.status_code == 200:
            return response.text

        elif response.status_code == 404:
            print(f"[404] Page not found: {url}")
            return None

        elif response.status_code == 403:
            print(f"[403] Blocked by server: {url}")
            return None
```

```
        elif response.status_code == 429:
            print(f"[429] Too many requests: {url}")
            return None

        else:
            print(f"[{response.status_code}] Unexpected error")
            return None

    except requests.exceptions.RequestException as e:
        print(f"[ERROR] Network issue: {e}")
        return None
```

🧠 **Notice this carefully:**

- Function **kabhi crash nahi karta**
- Failure me `None` return hota hai
- Decision **yahin** liya ja raha hai

🔑 IMPORTANT DESIGN RULE (Yaad rakhna)

> ❌ Parsing function ko error handle nahi karna ☑ Network function ko handle karna chahiye

Isliye:

- `fetch_html()` → decision maker
- `parse_html()` → sirf parsing

🤔 THINKING EXERCISE (Answer honestly)

1️⃣ `fetch_html()` ne `None` return kiya → `parse_html(None)` call karna chahiye ya nahi?

2️⃣ 403 aane par **retry karna** sahi hai ya galat? Kyun?

3️⃣ 429 aane par **sleep kahan hona chahiye**?

- `fetch_html()` ke andar?
- ya main loop me?

4️⃣ Agar ek page fail ho jaaye, kya **poora scraper rukna chahiye**?

🧪 DEBUGGING TASK (VERY IMPORTANT)

❌ Buggy code (REAL LIFE BUG)

```
html = self.fetch_html(current_url)
soup = self.parse_html(html)
self.extract_records(soup)
```

❓ Questions:

- Agar `html = None` ho gaya toh?
- Crash kahan hoga?
- Isko **minimum lines me** kaise safe banaoge?

✍️ Sirf socho. Code abhi mat likho.
