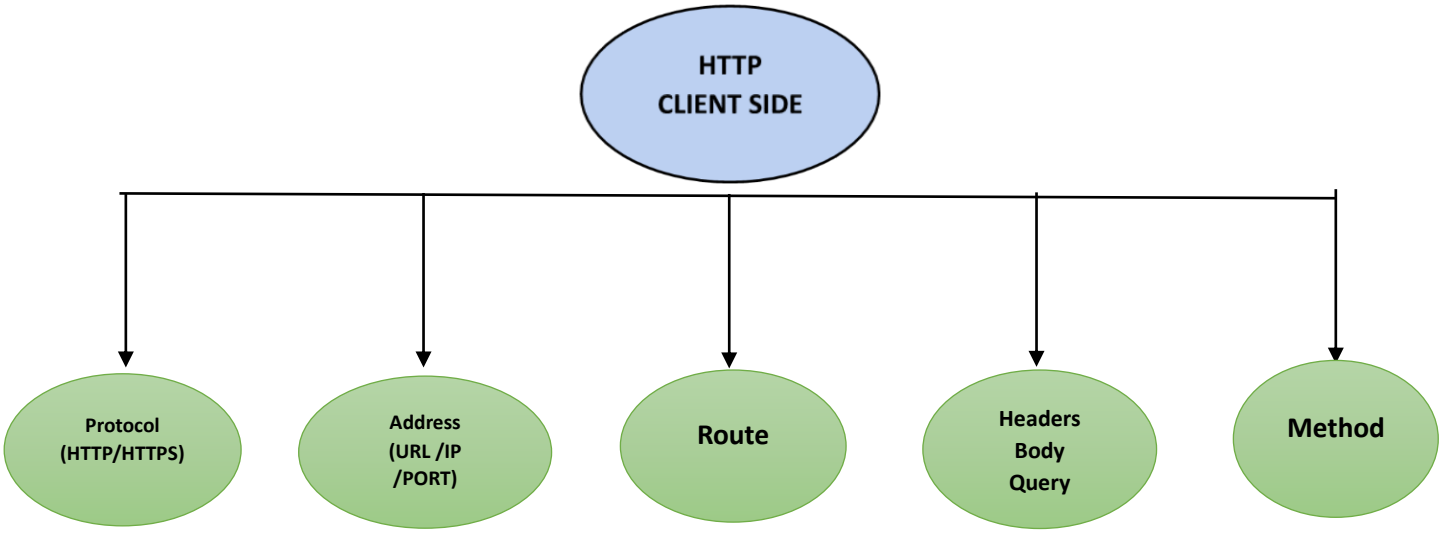
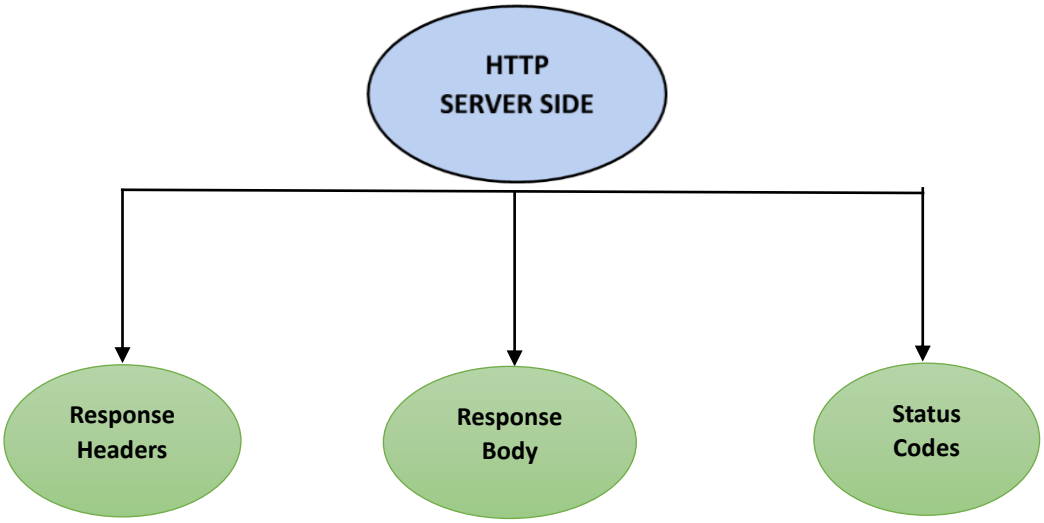
HTTP PROTOCOL

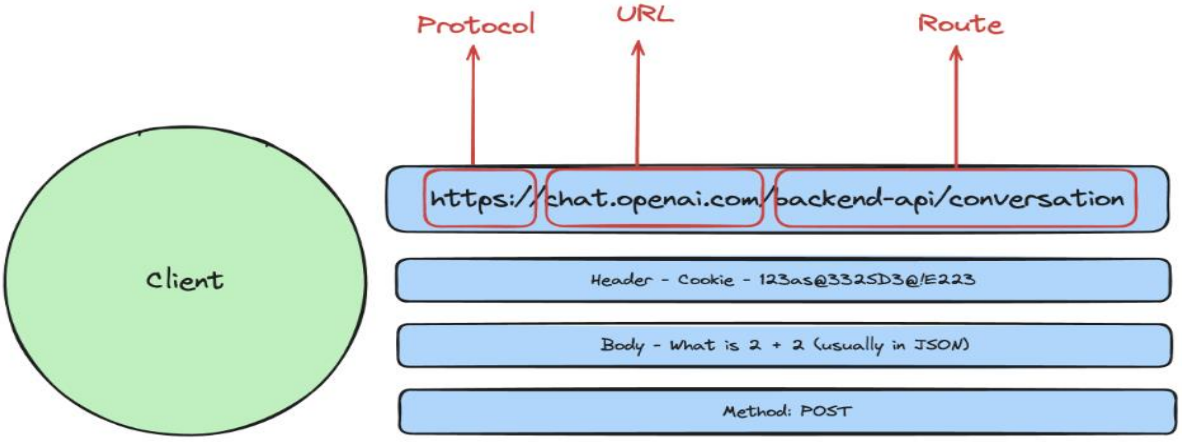
- Things client needs to worry about

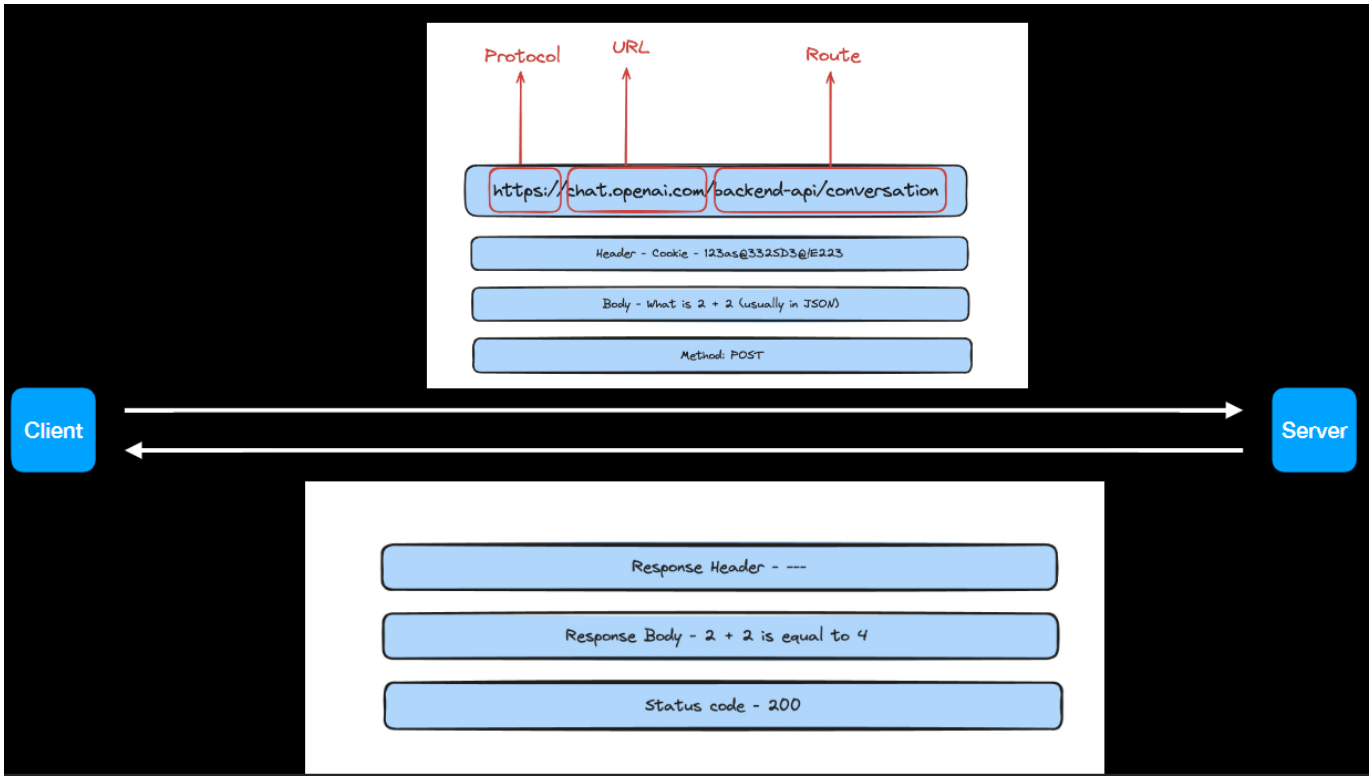


- Things server needs to worry about

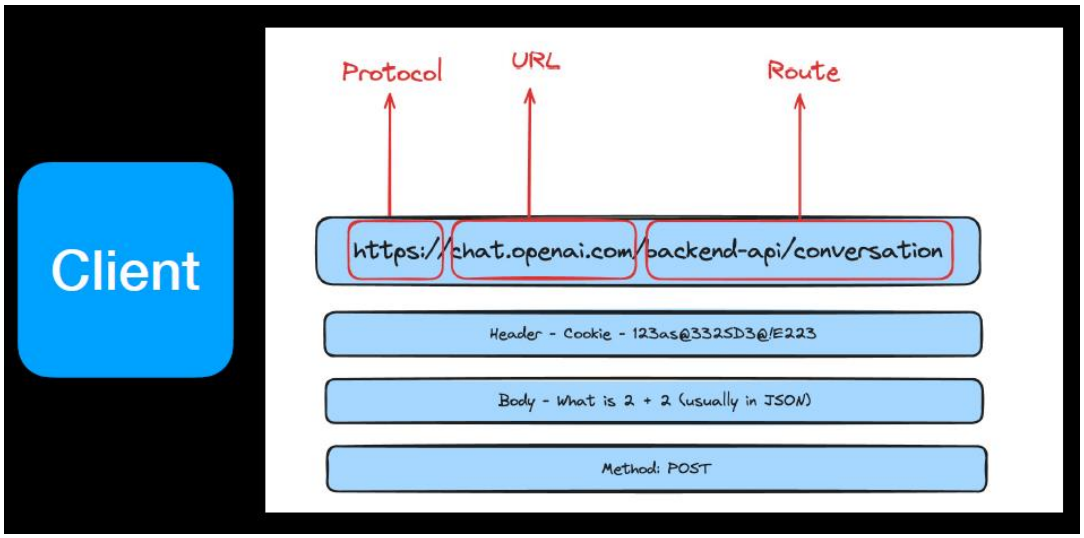


- Usually, Communication would happen as shown below





What happens in your browser after you fire a request (typically an URL)



1. **Browser Parses the URL:**

- When you type a web address (like "google.com") into the browser or click a link, the browser breaks down the address into parts to understand what you're asking for

  scheme://host:port/path?query#fragment

  - **Scheme:** It figures out if it should use "http" or "https" (secure) for the communication.
  - **Host:** It identifies the server you want to talk to, like "google.com."

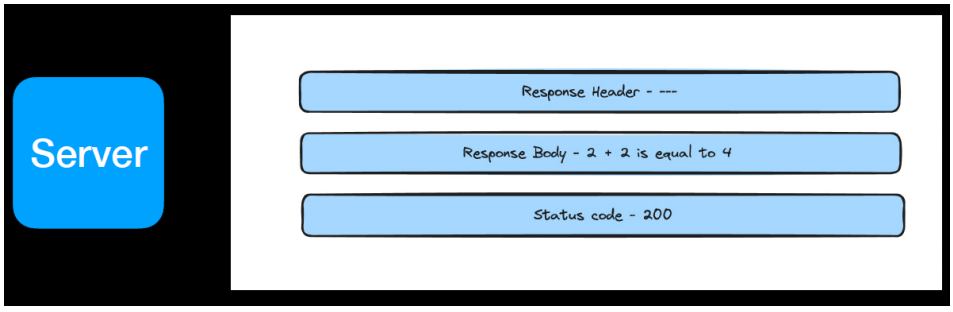
- **Port**: If specified, it's like a door number on the server. Usually, it's 80 for regular communication and 443 for secure communication.
- **Path**: It pinpoints the specific thing you want from the server, like a particular page.
- **Query Parameters**: If there are any extra instructions for the server, like searching for something specific.
- **Fragment**: A part of the page you want to go directly to.

2. **DNS Lookup (Converts google.com to an IP)**
   - The DNS is a distributed system that translates human-readable domain names (like "google.com") into IP addresses that machines can use for communication.

   - The browser needs to find the actual location (IP address) of the server you want to talk to:
     - **Local Cache**: It checks if it already knows the IP address from before.
     - **Operating System Cache**: If not, it asks the computer's memory if it knows.
     - **Recursive DNS Servers**: If still unsure, it asks special servers on the internet that are really good at finding IP addresses.
     - **Authoritative DNS Servers**: Finally, it asks the official servers that know all the IP addresses.

3. **Establishes a Connection to the IP (Does Handshake)**
   - Once the browser has the IP address of the server, it establishes a connection to that server. This involves a process known as the three-way handshake.
     - **SYN (Synchronize)**: It's like saying, "Hey, can we talk?"
     - **SYN-ACK (Synchronize-Acknowledge)**: The server replies, "Sure, let's talk."
     - **ACK (Acknowledge)**: The browser says, "Great, we're talking now!"
   - **Establishing a Connection**:
     - This process ensures that both the client (browser) and server agree to establish a reliable connection before any data transfer occurs.

What happens on your server after the request is received



1. **You Get the Inputs (Route, Body, Headers)**

   Imagine you have a calculator server, and someone sends you a request. This request is like a note that contains instructions and numbers.

   - **Route (Path)**: The route is like telling you which function to use on the calculator. For example, if the route is "/add," it means the person wants to add numbers.
   - **Body**: The body is like the actual numbers or data that the person wants you to use in the calculation. If they want to add 5 and 3, the body might say "5 + 3."
   - **Headers**: Headers are like additional notes. Maybe there's a note saying, "I prefer the answer in a special format."

2. **You Do Some Logic on the Input, Calculate the Output:**
   - **Route Processing**: You look at the route and say, "Ah, they want to add numbers." If it was "/subtract," you'd know they want to subtract.
   - **Body Calculation**: You take the numbers from the body and perform the requested operation. In our case, you add 5 and 3, getting 8.
   - **Logic**: Depending on the situation, you might do some extra logic. For example, you could check if the numbers are valid or handle special cases.

3. **You Return the Output (Body, Headers, and Status Code):**

   After doing the calculation, you prepare the response to send back to the person:

   - **Output Body**: This is the result of the calculation. In our case, it's the number 8.
   - **Headers**: If there were any special notes in the headers, you make sure to include them in your response.
   - **Status Code**: This is like a quick note at the top of your response, saying if everything went well or if there was a problem. If the calculation was successful, you might use a status code like **200 (OK)**. If there was an issue, you'd use a different code, like **400 (Bad Request)**.