Copilot

When it comes to **HTTP status codes**, both **400** and **422** are used to indicate client errors, but they serve different purposes:

1. **400 (Bad Request)**:
    - The server responds with a **400 status code** when it cannot or will not process the request due to something perceived as a **client error**.
    - Examples of scenarios that trigger a **400** include:
        - **Malformed request syntax**: When the client sends a request with incorrect syntax.
        - **Invalid request message framing**: If the request message structure is incorrect.
        - **Deceptive request routing**: When the server detects deceptive routing.
    - In essence, **400** signifies that the server encountered issues related to the request itself.
2. **422 (Unprocessable Entity)**:
    - The **422 status code** indicates that the server understands the content type of the request entity and that the syntax of the request entity is correct.
    - However, the server was **unable to process the contained instructions** due to semantic errors.
    - For instance, this error might occur if an XML request body contains well-formed (syntactically correct) but semantically erroneous XML instructions.
    - **422** is more focused on semantics and is defined in RFC 4918, which is a WebDAV extension for HTTP/1.1.

Now, let's consider the client error scenarios you mentioned:

1. **Invalid parameter**: If the client provides invalid parameters (e.g., wrong data type or nonexistent user ID), both **400** and **422** could apply. However, **400** is more general, while **422** emphasizes semantic correctness.
2. **Missing required parameters**: Again, both codes could be used, but **422** aligns better with semantic correctness.
3. **Hashing failure**: This could be considered a semantic error, making **422** a suitable choice.
4. **Blocked content**: Since this involves processing instructions, **422** is appropriate.
5. **Deleting an entry still in use**: This scenario also relates to processing instructions, favoring **422**.
6. **Signature mismatch**: Semantic correctness points to **422**.
7. **Attribute limit reached**: **422** fits well here too.

In practice, both **400** and **422** are valid choices, but if you want to emphasize semantic correctness, **422** is a better fit. Keep in mind that some APIs (like Google) use **400**, while others (like Twitter) opt for **422**. Ultimately, consistency within your application matters most! 🤓👍