A black screen with white text

Description automatically generated

A computer screen with green and white text

Description automatically generated

**Advantages of using Kwargs**:

1. Flexibility to Accept Variable Arguments: Kwargs allow you to pass a variable number of keyword arguments to a function, allowing it to be flexible when handling inputs without predefining them.
2. Simplifies Handling Optional Parameters: Kwargs make defining optional parameters easy by gathering them in kwargs instead of explicitly defining optional parameters.
3. Easily Extensible Functions: When using Kwargs, you can easily extend the functions functionality without modifying the function signature.
4. Interfacing with APIs or Wrapping Functions: When working with external APIs or when creating wrapper functions, kwargs allow the user to pass arguments dynamically. This makes it easier to handle API responses.
5. Clean code with Default Values: Using Kwargs, you can provide default values using .get() which leads to cleaner code.

**Disadvantages of using Kwargs**:

1. Lack of Explicitness: Since parameters are not explicitly listed in the functions signature, it can make the function harder to understand.
2. Potential for Errors: The lack of Explicit names can lead to mistakes that may not trigger until runtime.
3. Harder to Maintain and Debug: When using Kwargs, it is more challenging to trace bugs due to the functions ability to accept any number of keyword arguments, this makes it harder to validate inputs, making debugging much more difficult.
4. Increased Complexity for Type Checking: Kwargs can complicate type checking, whether manually or using static type checkers, kwargs can complicate their ability to check code.
5. Overuse can lead to Poor API Design: Overuse of Kwargs can lead to confusing and hard to read ApIs.