**Doxygen Commenting Guidelines**

To maintain consistency, readability, and clarity in the documentation generated using Doxygen, please follow these guidelines when adding comments to the codebase.

**1. Use Proper Doxygen Tags**

* Always use Doxygen's standard tags for documentation. The most commonly used tags are:
	+ @brief: A short description of a function or class.
	+ @param: Describes a function parameter.
	+ @return: Describes the return value of a function.
	+ @note: Additional important information.
	+ @warning: Highlights cautionary information.
	+ @todo: For marking incomplete sections or future improvements.

Example:



**2. Write Meaningful Descriptions**

* **@brief** should provide a concise overview.
* Follow up with additional details after @brief if necessary.
* Provide explanations for complex logic or algorithms within function documentation.
* Avoid redundant or overly verbose descriptions. Keep it simple but informative.

Example:

**3. Document All Public Functions and Classes**

* Every public function, class, method, and member variable must have a Doxygen comment. Include the purpose of the function, any parameters, and the expected return value.
* For class-level documentation, explain the class's overall purpose and important methods or members.

Example:



**4. Document Parameters and Return Values**

* Always use @param to describe all input parameters, specifying what they represent and any important constraints (e.g., "must be non-negative").
* Use @return to describe the function’s return value, even if it’s void or a simple type.

Example:



**5. Handle Edge Cases and Warnings**

* Use @warning or @note to indicate edge cases, potential pitfalls, or assumptions made by the function.
* Highlight performance issues or important conditions (e.g., if a function is not thread-safe).

Example:



**6. Keep Comments Up-to-Date**

* Always update comments when the code is modified. Outdated or incorrect comments can cause confusion and mislead users.
* Ensure that the comments accurately reflect the functionality and behavior of the code.

**7. Use Consistent Style**

* Use consistent formatting, indentation, and punctuation in Doxygen comments across the codebase.
* Write comments in full sentences with proper grammar and punctuation for readability.

Example:



**8. Private/Internal Code**

* If internal or private methods need to be documented for development purposes, mark them clearly as internal using @internal or @private so that they are not included in public-facing documentation.

Example:

