**Coding Standards and Guidelines**

**1. Introduction**

* **Purpose**: To ensure readability, consistency and cleanliness of code.
* **Scope**: All Coop projects

**General Principles**

* **Consistency**: Be consistent,
* **Readability**: Prioritize code clarity and understandability.
* **Performance**: Ensure code is optimized for efficiency and scalability.
* **Maintainability**: Write code that is easy to extend and debug.
* **Portability**: Do not introduce elements that are not portable.
* **Dead Code**: Dead code shall not be left inside the project.
* **Organization**: The MAIN file shall include a single include reference and only function calls from inside. Keep separate work separate. Declare types where they are used unless they are widespread like api,networking,core aka math functions file management.

**Naming Conventions**

* **Variables**:
  + Use descriptive names and (camelCase).
* **Functions/Methods**:
  + Use PascalCase Or underscores
* **Constants**:
  + Use ALL\_CAPS with underscores for constants.

**5. Code Formatting**

* **Indentation**:
  + Properly indent all code as if it were python
  + All types and functions must have at least 1 return separating them.
  + All code inside a function or type must be tabbed
* **Line Length**:
  + Do not exceed the editors window with aka scrolled
* **Comments**:
  + // for comments do not use the tilde key. Or /\* \*/

**6. Coding Practices**

* **Error Handling**:
* Test and debug your own code before including it in the project don’t make messes for others.
* **Code Reuse**:
  + Avoid duplication; encourage modular design.
* **Dependencies**:
  + Avoid hard coding and encourage dependency injection.
* **Optimization**: Always keep in mind the impact you are having, on low end systems program as if you have limited resources. Avoid loops and consolidate loops, if you can check 1 item per loop instead of looping through each item then you can combine loops into tha main loop and avoid bottlenecks in code. Loops in memblock and iterative code are the most waist full code there is.

**7. Language**

English is the most common known language in the world my native language and the language of tgc. This being said if you do not speak or write well in English plese try to program using full descriptions in your native language then convert it via translator. Better to be bloated then not understandable.

**8. Testing Standards**

* **Unit Testing**:
  + Project must be complied and all features tested before pushing to repo.
* **Testing**:
  + Testing, for pc, steam deck , controller support, networking, web integration, exploits, shaders,

**9. Version Control Guidelines**

* **Branching Strategy**:
  + Guidelines for main, feature, bugfix, and release branches.
* **Commit Messages**:
  + Yes descriptive of added or changed features and bug fixes..
* **Pull Requests**:
  + Review process, required approvals, and merge strategies.
* **Tagging and Releases**:
  + Versioning rules

**10. Documentation Standards**

* **Code Comments**:
  + Comment often.
* **API Documentation**:
  + Standards for writing and maintaining API references.
* **Project Documentation**:
  + README, changelog, and design documents.

**11. Code Review Process**

* **Review Checklist**:
  + Do not rewrite or remove others code, You may write your own and disable the other code if it is not functional, aka bugs. If there is concern we can take a vote to remove code due to perhaps portability or being non dynamic
* **Feedback**:
  + Providing constructive comments.

**Third-Party Libraries**:

If you implement a library you are responsible for itegration.