Acceptatiecriteria

Ready

- De Issue heeft een wireframe
- De Issue voldoet aan het voorgeschreven format (als – wil ik – zodat)
- De Issue heeft één of meer van de afgesproken labels (bug, feature, critical)

Done

- De Issue is getest
- De Issue Heeft geen bugs
- De Issue is goedgekeurd door de product owner.
- De code log moet bij de issue staan

Code conventions

- NAMING/CASING:
  - Use Pascal case (e.g. ExamplePlayerController, MaxHealth, etc.) unless noted otherwise
  - Use camel case (e.g. examplePlayerController, maxHealth, etc.) for local/private variables, parameters.
  - Avoid snake case, kebab case, Hungarian notation
  - If you have a Monobehaviour in a file, the source file name must match. Use nouns for variable names: Variable names should be clear and descriptive because they represent a specific thing or state.
- CLASSES or STRUCTS:
  - Name them with nouns or noun phrases.
  - Avoid prefixes.
- Prefix Booleans with a verb: These variables indicate a true or false value. Often they are the answer to a question, such as: Is the player running? Is the game over?
  - Prefix them with a verb to make their meaning more apparent. Often this is paired with a description or condition, e.g., isDead, isWalking, hasDamageMultiplier, etc.

- Use meaningful names. Don't abbreviate (unless it's math): Your variable names will reveal their intent. Choose names that are easy to pronounce and search for.
  - While single-letter variables are fine for loops and math expressions, don't abbreviate in other situations. Clarity is more important than any time saved from omitting a few vowels.
  - For quick prototyping, you can use short "junk" names temporarily, and then refactor to more meaningful names later on.
- Use pascal case for public fields, and use camel case for private variables

- C# definition : https://learn.microsoft.com/en-us/dotnet/csharp/fundamentals/coding-style/coding-conventions

Gitlab: https://gitlab.rijnijssel.nl/00318282/virtual-reality-game