**Lazy vs Lateinit**

**Lazy**:

Initialization: Delayed initialization of a property until its value is accessed for the first time.

Syntax: Implemented using the lazy property delegate.

Use Case: Suitable for properties whose values might not always be needed and are potentially expensive to compute.

Initialization: The value is computed only once, when it is first accessed. Subsequent accesses return the cached value without recomputing it.

Thread Safety: Supports different thread-safety modes (SYNCHRONIZED, PUBLICATION, NONE) to cater to various concurrency needs.

**Lateinit:**

Initialization: Postpones the initialization of a non-null property until it is actually used.

Syntax: Declared using the lateinit modifier.

Use Case: Appropriate for mutable properties that need to be initialized after object creation but before being used.

Initialization: The property must be initialized before accessing it; otherwise, a lateinit property access exception is thrown.

Nullable Types: Can't be used with nullable types.