

FREE CHECKLIST

Production Code Review Checklist for Python and Java

Forty questions to ask of any change before it ships, grouped by where production bugs actually hide: money, time, text, errors, collections and state. Written for working developers and career switchers; every item has a one-line fix.

HOW TO USE IT

- Review the diff, not the description. Open the changed files and answer each question that applies.
- A "no" is a comment, not a block. Most items are a two-line fix; say which line and why.
- Keep the review under thirty minutes. If the checklist finds more than five problems, the change is too big: ask for a split.
- Items with a ► are the ones covered in the Coding Solutions short videos.

1. MONEY AND NUMBERS

- ► Is money ever a float ? `int(4.35 * 100)` is 434, not 435. Use `Decimal` (Python) or `BigDecimal` built from a *string* (Java), and store cents as integers.
- Does any `BigDecimal` use `equals` ? `2.50` does not equal `2.5` ; compare with `compareTo` .
- Is an ID or a large count parsed as a float anywhere (JSON, CSV, a spreadsheet export)? Integers above 2^{53} lose digits.
- Does every division say what happens at zero, and is integer division intended where it appears?
- Are percentages and rates stored in one unit (0.07 or 7, never both)?
- Is rounding done once, at the end, with a stated rounding mode, rather than at every step?

2. DATES AND TIME

- ► Does any Java date pattern use `YYYY` ? That is the week-based year: 29 Dec 2025 prints as 2026. Use `yyyy` .
- ► Does the code read `timedelta.seconds` ? It drops the days. Use `total_seconds()` .
- Are naive and aware datetimes ever mixed? Subtracting one from the other raises; store everything in UTC with a timezone attached, and convert at the edge.
- Is `now()` called in more than one place in a single operation? Capture it once and pass it down, or two timestamps that should match will not.
- Does month arithmetic clamp? 31 Jan plus one month is 28 Feb, and adding another month from there drifts. Add months from the original date.
- Is anything serialised to JSON with a datetime in it, and does the serialiser know how?

3. ERRORS AND LOGGING

- ► Is there an `except Exception: pass` or an empty `catch` ? It hides typos as well as outages. Catch the narrowest type you can name, and log what you skip.
- Does every caught exception either recover, re-raise, or log with enough context to find the record (an ID, not just "failed")?
- Are log messages built with f-strings before the level check? The string is built even when the level is off. Pass arguments to the logger instead.
- Is `logging.info` relied on at the default level? It is silent until the logger is configured.
- Does a retry loop have a maximum, a backoff and an idempotent body?

4. TEXT, PARSING AND INPUT

- Is user input validated at the boundary and trusted after it, rather than re-validated in five places?
- Does `split` drop trailing empty fields (Java) or produce an empty string that is later converted to a number?
- Are JSON object keys assumed to stay integers? They come back as strings.
- Is any string compared for equality where case, whitespace or Unicode normalisation matters?
- Does a regular expression anchor both ends, and is it compiled once?
- Is the size of a file, request body or upload bounded before it is read into memory?

5. COLLECTIONS AND STATE

- ► Does anything depend on the order of a `set`, a `HashMap` or a `HashSet`? There is none you can rely on. Use `sorted`, `dict.fromkeys` or a `LinkedHashMap`.
- Is a list modified while it is being iterated?
- Is a mutable object shared where a copy was intended (a default argument, a class attribute, a cached list)?
- Does `Collectors.toMap` handle duplicate keys, or will the second one throw?
- Is `Arrays.asList` given a primitive array (one element, the array itself) or later resized (it cannot be)?
- Does `is` (Python) or `==` on boxed types (Java) compare identity where equality was meant?

6. CONTROL FLOW

- ► Can an `if / elif` chain with overlapping ranges match the wrong branch? Test the most specific condition first.
- ► Is a condition always true because two comparisons are joined with `or` when `and` was meant?
- Is a counter or accumulator reset inside the loop that should accumulate it?
- Does a function that should return in every branch have a branch that falls off the end?
- Is a field accessed through a parent-typed reference where the subclass value was expected (Java fields are not polymorphic)?

7. BEFORE YOU APPROVE

Tests

- Is there a test for the bug this change fixes, and does it fail on the old code?
- Do tests use fixed dates, fixed random seeds and no network?
- Does one test cover the boundary (empty, zero, one, maximum)?

Operations

- Can this change be rolled back without a data migration?
- Is every new secret read from the environment, not the code?
- Will the on-call engineer understand the new log lines at 3 am?

Want a second pair of eyes on a real change?

Coding Solutions does architecture and code reviews for Singapore teams: an external read on your system or a module, with the risks ranked and the fixes sequenced, in one to two weeks. Details at codingsolutions.dev/consulting, or message [@IsaacNgCS](https://t.me/IsaacNgCS) on Telegram.