Bandit Python security linter

CONCEPTS

> Every finding has a severity (impact) and a confidence (how sure Bandit is). Triage high/high first.

Python files parsed to AST
↓ plugins
Tests B1xx-B7xx selected by -t / -s
↓ filters
Issues severity + confidence
↓ -f
Report screen, json, html...

> Bandit exits 1 when it reports issues, so it works as a CI gate; --exit-zero overrides that.

SCANNING

`bandit -r <dir>`
Scan a directory recursively

`bandit <file.py> ...`
Scan specific files

`bandit --version`
Show Bandit and Python version

`bandit -r <dir> -x <paths>`
Exclude comma-separated paths

`bandit -r <dir> -v`
Verbose: list files scanned and excluded

`bandit -r <dir> --exit-zero`
Always exit 0

FILTERING

`bandit -r <dir> -l / -ll / -lll`
Severity low / medium / high and up

`bandit -r <dir> -i / -ii / -iii`
Confidence low / medium / high and up

`bandit -r <dir> -t <ids>`
Run only these test IDs

`bandit -r <dir> -s <ids>`
Skip these test IDs

OUTPUT

`bandit -r <dir> -f json`
Machine-readable report

`bandit -r <dir> -f <format>`
csv, html, json, screen, txt, xml, yaml, custom

`bandit -r <dir> -o <file>`
Write the report to a file

CONFIGURATION

`bandit -r <dir> -c <file>`
Use a config file

pyproject.toml

```
[tool.bandit]
exclude_dirs = ["tests", ".venv"]
skips = ["B101"]
```

SUPPRESSING

`# nosec`
Ignore all findings on this line

`# nosec B602`
Ignore only the named test(s)

`bandit -r <dir> --ignore-nosec`
Report lines marked nosec anyway

`bandit -r <dir> -b <baseline.json>`
Show only issues not in the baseline

COMMON FINDINGS

Tests you will meet most

ID	Flags
B101	assert (stripped under -O)
B105-B107	hardcoded password
B108	hardcoded /tmp path
B113	request without timeout
B301	pickle load
B311	random for security use
B324	weak hash (MD5, SHA1)
B506	yaml.load unsafe
B602	subprocess with shell=True
B608	SQL built from strings