The 7-step application risk assessment checklist

1. Identify the application's context

- [ ] **Map the architecture:** document APIs, databases, microservices, hosting and third-party services.
- [ ] **Trace data flows:** draw how data moves between the application and other systems, and mark the trust boundaries.
- [ ] **Classify data and users:** list user roles and access levels, and rate the data's sensitivity by its impact on confidentiality, integrity and availability (the CIA triad).

Done when: you have an up-to-date architecture diagram, a data-flow diagram and a data classification for the application.

2. Find known vulnerabilities

- [ ] **Scan the application and its hosts:** use a dynamic scanner such as ZAP for the web front end, and an infrastructure scanner or agent for the servers and endpoints it runs on.
- [ ] **Check against vulnerability databases:** match findings to CVE (Common Vulnerabilities and Exposures) records in the NVD.
- [ ] **Check patch status:** confirm which OS and third-party patches are missing, and which vulnerabilities have no patch yet.

Done when: every known vulnerability in the application and its hosts is listed with its CVE, and its patch status.

3. Score likelihood with threat intelligence

- [ ] **Separate severity from likelihood:** CVSS measures how severe a vulnerability is. For how likely it is to be exploited, add **EPSS** scores and check **CISA's Known Exploited Vulnerabilities (KEV)** catalog.
- [ ] **Add threat context:** use threat intelligence feeds to see which vulnerabilities are being exploited in the wild, and by whom.
- [ ] **Check reachability:** confirm whether a vulnerable component is actually exposed or reachable in your environment.

Done when: each finding has a likelihood rating based on exploit activity, not just its CVSS score.

4. Map compliance and regulatory requirements

- [ ] **Map requirements:** for example, PCI DSS for payment data, HIPAA for health data, GDPR for personal data.
- [ ] **Evaluate controls:** check existing controls against frameworks such as NIST CSF, ISO 27001 or CIS Benchmarks.
- [ ] **Verify audit trails:** make sure the application logs enough activity to support audits and investigations.

Done when: you know which requirements apply, and which controls are missing.

5. Review history and configuration drift

- [ ] **Review past incidents:** look at security incidents involving this application, or similar ones, to spot repeat weaknesses.
- [ ] **Track trends:** watch how many vulnerabilities appear and how long they take to fix over time.
- [ ] **Detect configuration drift:** compare current settings against your secure baseline.

Done when: recurring issues and drift from the baseline are documented.

6. Test the code and its components

- [ ] **Static and dynamic testing:** use SAST to find flaws in source code, and DAST to find flaws in the running application.
- [ ] **Software composition analysis (SCA):** use an SCA tool such as Snyk or OWASP Dependency-Check to find vulnerable or outdated open-source libraries, and keep a software bill of materials (SBOM).
- [ ] **Runtime protection:** consider runtime application self-protection (RASP) or exploit-prevention tools for critical applications that can't be patched quickly.

Done when: code, dependencies and runtime have each been tested, and the findings are added to your list.

7. Agree on risk and remediation with stakeholders

- [ ] **Run a cross-functional review:** bring together security, IT operations, developers and the application owner.
- [ ] **Score and rank:** combine likelihood and impact into one risk score per finding, and agree on a fix-by date for each risk level.
- [ ] **Define remediation workflows:** decide who fixes what, and how: patch, configuration change, compensating control, or accepted risk with sign-off.

Done when: every high and critical risk has an owner, a fix-by date and an agreed remediation method.

Risk scoring grid

Risk score = likelihood × impact

Use a 1–5 scale for each factor. Agree on rating definitions and priority thresholds with your team before scoring.

Rating	Likelihood	Impact
1	Rare	Negligible
2	Unlikely	Minor
3	Possible	Moderate
4	Likely	Major
5	Almost certain	Severe

Base likelihood on exploit activity, reachability and existing controls. Base impact on data sensitivity, business disruption and compliance exposure.

Likelihood ↓ / Impact →	1	2	3	4	5
1	1	2	3	4	5
2	2	4	6	8	10
3	3	6	9	12	15
4	4	8	12	16	20
5	5	10	15	20	25

Agreed priority thresholds

Low: _____ Medium: _____ High: _____ Critical: _____

Complete one entry per finding.

Finding / CVE	Affected asset	Likelihood (1–5)	Impact (1–5)	Risk score	Owner