MEV

TECHNICAL DUE DILIGENCE CHECK LIST

- Software Development Audit Checklist
- Cloud Infrastructure Audit Checklist
- Advanced Checks

SOFTWARE DEVELOPMENT AUDIT CHECKLIST

1. Code Quality

- [ ] Are coding standards clearly defined and documented?
- [ ] Are coding standards consistently followed?
- [ ] Are peer code reviews conducted for all code changes?
- [ ] Are automated tools (e.g., linters, static analysis) in use?
- [ ] Is there a process for refactoring legacy code?

2. Testing

- [ ] Are unit tests written for all new features?
- [ ] Are integration tests in place?
- [ ] Is there a comprehensive test suite covering critical functionality?
- [ ] Are tests automated, and do they run as part of the CI/CD pipeline?
- [ ] Is test coverage monitored and improved where necessary?

3. Version Control

- [ ] Is version control software (e.g., Git) in use?
- [ ] Is a branching strategy (e.g., Gitflow) clearly defined and followed?
- [ ] Are code commits tagged with meaningful messages?
- [ ] Are code reviews mandatory before merging to main branches?
- [ ] Are merges and releases automated using CI tools?

4. CI/CD

- [ ] Is there a CI/CD pipeline in place for the project?

- [ ] Are builds and tests automated?
- [ ] Is deployment automated across all environments (e.g., dev, staging, production)?
- [ ] Is rollback of failed deployments automated?
- [ ] Are CI/CD pipelines monitored for failures?

5. Documentation

- [ ] Is the codebase properly documented (e.g., comments, README files)?
- [ ] Is documentation available for onboarding new developers?
- [ ] Are API specifications documented (e.g., OpenAPI, Swagger)?
- [ ] Is there user-facing documentation (e.g., manuals, help guides)?
- [ ] Is documentation updated alongside code changes?
- [ ] Are tools in place for automated documentation generation?

6. Architecture

- [ ] Is the architecture documented and shared with the team?
- [ ] Are design patterns and principles followed (e.g., SOLID, DRY)?
- [ ] Is the architecture modular and loosely coupled?
- [ ] Is the system scalable to handle future growth?
- [ ] Are architectural changes reviewed and validated by the team?
- [ ] Is the system architecture using purpose-fit data stores?

7. Performance

- [ ] Are performance baselines defined for critical components?
- [ ] Is performance testing conducted (e.g., load, stress, and endurance tests)?
- [ ] Are performance issues logged and prioritized?
- [ ] Are tools used for performance monitoring in production?
- [ ] Are performance improvements planned and implemented iteratively?

8. Security

- [ ] Are secure coding practices followed (e.g., input validation, encryption)?
- [ ] Are dependencies regularly scanned for vulnerabilities?
- [ ] Is there a process for managing security incidents?
- [ ] Are authentication and authorization implemented correctly?
- [ ] Are penetration tests conducted regularly?
- [ ] Are data exfiltration prevention measures in place (e.g. DLP)?
- [ ] Are code repositories regularly scanned for hard-coded secrets?
- [ ] Is there a password policy in place?
- [ ] Are service account credentials regularly rotated?
- [ ] Is all data-at-rest encrypted (AES/RSA with min 256 bits)?
- [ ] Is all data-in-transit encrypted (TLS v1.2 or higher)?
- [ ] Are security events logged and monitored (e.g. failed auth attempts)?
- [ ] Is there antivirus and antimalware software running on development machines?
- [ ] Is there antivirus and antimalware software running on production hosts?
- [ ] Are dependencies vendored to prevent supply chain attacks?

9. Team Collaboration

- [ ] Are collaboration tools (e.g., Jira, Slack, Confluence) in place?
- [ ] Are regular team meetings (e.g., stand-ups, retrospectives) conducted?
- [ ] Is knowledge sharing encouraged and practiced?
- [ ] Are roles and responsibilities clearly defined?
- [ ] Are new team members onboarded effectively?
- [ ] Do all team members actively participate in code reviews?
- [ ] Are there upskilling/training opportunities available?

10. User Feedback

- [ ] Is user feedback collected systematically (e.g., surveys, analytics)?
- [ ] Is feedback reviewed and acted upon regularly?
- [ ] Are user feedback loops integrated into the development process?

- [ ] Are user pain points tracked and prioritized for resolution?
- [ ] Is telemetry data used to improve user experience?

11. Maintainability

- [ ] Is the codebase easy to read and understand?
- [ ] Are variables, functions, and classes meaningfully named?
- [ ] Is the software broken into manageable modules or components?
- [ ] Are components loosely coupled and highly cohesive?
- [ ] Is technical debt tracked and prioritized for resolution?
- [ ] Are dependencies regularly updated and compatible?
- [ ] Are all dependency versions locked?
- [ ] Are APIs properly versioned?
- [ ] Are there documented procedures for handling breaking changes?
- [ ] Is the software using an ORM?
- [ ] Is the software managing data model changes through a migration tool?

12. Scaling

- [ ] Is the architecture designed for horizontal and vertical scaling?
- [ ] Are microservices or modular designs used where appropriate?
- [ ] Are databases optimized for scaling (e.g., sharding, partitioning)?
- [ ] Are load and stress tests conducted regularly?
- [ ] Are performance bottlenecks identified and resolved proactively?
- [ ] Are system throughput limits well understood and monitored?

CLOUD INFRASTRUCTURE AUDIT CHECKLIST

1. Compute Resources: Instance Management

- [ ] Are instances configured for automated updates and patch management (e.g., AWS Systems Manager Patch Manager) or is formal policy and procedure documented and followed?
- [ ] Are there any instances that are using old or outdated AMIs that need upgrading or patching?

2. Compute Resources: Health Checks & Monitoring

- [ ] Are health checks (e.g., instance status checks, load balancer health checks) configured and regularly monitored?
- [ ] Are instance performance metrics (CPU, memory, disk) monitored?
- [ ] Are auto-scaling policies in place and functioning correctly for instances requiring scaling?

3. Compute Resources: Security

- [ ] Are IAM roles applied for instances to ensure the least privilege principle is followed?
- [ ] Are instances accessible only through approved mechanisms (e.g., AWS Systems Manager instead of direct SSH access)?

4. Infrastructure: Infrastructure Inventory

- [ ] Is a complete list of infrastructure components (EC2, RDS, EKS, etc.) available and verified?

- [ ] Is access to these infrastructure components functional and verified?
- [ ] Are cloud provider limits and quotas known and documented?

5. Infrastructure: Infrastructure Management

- [ ] Are the present infrastructure management approaches and tools known (e.g., Infrastructure as Code using Terraform, CloudFormation)?
- [ ] Is the component scaling approach documented and understood (e.g., autoscaling for EC2, scaling for Kubernetes clusters)?

6. Infrastructure: Mission-Critical Resources

- [ ] Is the list of mission-critical resources clearly defined and known by the team?
- [ ] Are any single points of failure identified, and is redundancy in place for critical resources?

7. Infrastructure: Network Design

- [ ] Are virtual networks (VPCs, subnets) properly segmented for security (e.g., separate public and private subnets)?
- [ ] Are VPC Peering, VPNs, or Transit Gateways optimally configured for inter-cloud communication?

8. DNS Management: Domain Registration & Maintenance

- [ ] Is the domain registration expiration date known, and is auto-renewal configured?
- [ ] Is the monitoring system configured to raise alerts when the domain is about to expire?
- [ ] Are all project domains listed and verified for accuracy?

9. CI/CD Pipeline: CI/CD Setup

- [ ] Are CI/CD pipelines set up for all deployable components of the system (e.g., Jenkins, GitLab CI, AWS CodePipeline)?
- [ ] Are quality gates (e.g., unit tests, code linting) correctly configured and enforced as part of the pipeline?

10. CI/CD Pipeline: Build Artifacts

- [ ] Are build artifacts stored separately from the runtime environment (e.g., using AWS S3, Azure Blob Storage)?
- [ ] Is artifact versioning and access controlled for CI/CD pipeline outputs?

11. CI/CD Pipeline: Deployment Process

- [ ] Is the CI/CD system integrated with infrastructure to handle deployments securely (e.g., AWS CodeDeploy, Azure DevOps Pipelines)?
- [ ] Are deployment rollback procedures implemented and tested?

12. Infrastructure Monitoring

- [ ] Is infrastructure monitoring configured to track resource utilization (CPU, memory, network, etc.)?
- [ ] Is black-box (endpoint) monitoring in place to ensure the uptime of critical services?

13. Application Monitoring

- [ ] Are application errors being tracked and logged (e.g., using Sentry, AWS X-Ray)?
- [ ] Is application performance monitoring (APM) configured, and are latency and throughput metrics tracked?

14. Security Monitoring

- [ ] Are alerts in place for critical security-related configuration changes (e.g., changes to IAM policies, security groups)?
- [ ] Are logs from services and applications forwarded to centralized logging services (e.g., CloudWatch, Log Analytics)?
- [ ] Are alerting mechanisms in place for anomalies or breaches (e.g., failed login attempts, high CPU usage)?
- [ ] Are escalation policies and procedures well-documented for responding to critical alerts?

15. 3rd-Party Monitoring

- [ ] Are 3rd-party dependencies monitored for availability and performance (e.g., ‘monitoring external APIs or services)?
- [ ] Are performance testing scenarios in place, and has performance testing been conducted?

16. Log Collection

- [ ] Are logs from all critical services (e.g., EC2 instances, RDS databases) being collected and forwarded to a central log management service?
- [ ] Are sensitive data and personally identifiable information (PII) excluded from log messages?

17. Log Retention & Access

- [ ] Are logs stored with a retention period of at least one year or as per business and regulatory requirements?
- [ ] Does the development and operations team have access to the necessary logs for troubleshooting?

18. Audit Logging

- [ ] Are audit logs (e.g., CloudTrail logs in AWS, Activity Logs in Azure) configured, and is access to them restricted?

19. Backup Configuration

- [ ] Are backup targets clearly defined for all critical resources (e.g., databases, EBS volumes)?
- [ ] Is the backup process automated and regularly tested for completeness and accuracy?

20. Backup Security

- [ ] Are backups encrypted at rest and in transit to protect sensitive data?
- [ ] Is the ability to restore from backups verified, and are test restores regularly performed?

21. Backup Retention

- [ ] Is the backup retention period configured according to business and regulatory requirements?

22. Security: Firewall & Security Groups

- [ ] Are firewalls and security groups in place and correctly configured for all infrastructure components (e.g., EC2, ALB, ECS, RDS)?
- [ ] Are unnecessary or excessive permissions in security groups removed, following the principle of least privilege?

23. Security: Identity & Access Management (IAM)

- [ ] Is IAM configured according to best practices (e.g., roles instead of long-term access keys)?
- [ ] Are MFA policies enforced for critical accounts, including root users?
- [ ] Is the root user secured and not in regular use?

24. Security: Least Privilege Enforcement

- [ ] Are roles, policies, and permissions following the principle of least privilege, granting only the necessary access required for tasks?

25. Security: Secrets & Password Management

- [ ] Are all users and services using unique credentials?
- [ ] Is a password policy defined and enforced with regular password rotation?
- [ ] Are secrets managed securely (e.g., using AWS Secrets Manager, Azure Key Vault), and are secret rotation policies in place?

26. Security: Compliance & Auditing

- [ ] Are security analysis tools in use (e.g., AWS Inspector, Azure Security Center) for continuous vulnerability detection?
- [ ] Is there a defined update and patching procedure for infrastructure components?
- [ ] Are governance frameworks (e.g., AWS Organizations, Azure Policies) in place and enforced for resource control and accountability?

27. Security: Credential Rotation

- [ ] Is there a procedure for credential rotation, especially for sensitive keys or tokens used by infrastructure?

ADVANCED CHECKS

Team Procedures

1. Operational Procedures

- [ ] Is the preferred maintenance window time range defined for the team?
- [ ] Is the time zone difference between the client, customer, and engineering team documented?

2. On-Call & Incident Response

- [ ] Is the on-call schedule for engineers known, and are escalation procedures documented?
- [ ] Are deployment and rollback procedures reproducible and well-documented?
- [ ] Are system component owners and their responsibilities clearly defined?

3. Alerts & Notifications

- [ ] Are notification channels (e.g., Slack, PagerDuty) set for each alert type?
- [ ] Is the alert review process defined, and are on-call engineers aware of response procedures?

4. Team Access & Security

- [ ] Is there a defined procedure for granting and revoking access to resources for team members?

- [ ] Is the secrets sharing policy defined, and are appropriate tools (e.g., AWS Secrets Manager, HashiCorp Vault) provided for sharing sensitive information?

Cost Management

1. Storage Utilization

- [ ] Are there any unused or orphaned volumes (e.g., unattached EBS or unmanaged disks)?
- [ ] Are storage classes/tiering being used appropriately to balance performance and cost (e.g., using archive or infrequent access tiers)?
- [ ] Are lifecycle policies implemented to manage data retention and control costs?

2. Instance Sizing

- [ ] Are instance types appropriately chosen for the workload (general-purpose, compute-optimized, memory-optimized, etc.)?
- [ ] Are there any over-provisioned or underutilized instances that should be adjusted?
- [ ] Have right-sizing recommendations (e.g., AWS Trusted Advisor, Azure Advisor) been considered and acted upon?

3. Tagging & Resource Management

- [ ] Are resources tagged consistently for tracking purposes (e.g., by department, project, or environment)?
- [ ] Are unused or orphaned resources regularly identified and cleaned up?

4. Cost Tracking

- [ ] Are cost monitoring tools (e.g., AWS Cost Explorer, Azure Cost Management) actively used to track and predict spending?
- [ ] Are budgets and alerts set to notify stakeholders of cost overruns?

Compliance & Auditing

1. Compliance Frameworks

- [ ] Are compliance frameworks such as GDPR, HIPAA, SOC 2 actively adhered to?
- [ ] Are cloud compliance checks automated (e.g., AWS Config Rules, Azure Policy)?

ABOUT US

MEV builds software—and we've been doing it for nearly 20 years.

Our approach is practical and straightforward, diving deep to understand your priorities and challenges to deliver worthy results.

We drive growth with the right combination of experience, proven technology, and product leadership.

For Mergers & Acquisitions (M&A), this proprietary checklist developed by our Principal Software Engineer and Director of Infrastructure provides the insights needed to make informed decisions without the guesswork.

+1 212-933-9921

solutions@mev.com

mev.com

linkedin.com