QuickKart Monitoring and Maintenance Checklist

Section 1: Monitoring Plan

1. Critical System Metrics and Thresholds

Metric	Description	Threshold	Purpose
System Uptime	Percentage of total time the application is operational	> 99.9%	Ensure continuous availability for customers
API Response Time	Average time taken for API endpoints to respond	< 300 ms	Maintain smooth customer interactions
Error Rate	Percentage of failed transactions or API calls	< 2% per hour	Detect application or integration issues early
CPU Usage	Average CPU utilization on servers	< 80%	Prevent performance degradation
Memory Usage	Percentage of memory used by application processes	< 75%	Avoid application crashes due to memory exhaustion
Database Connection Pool Usage	Percentage of active connections	< 85%	Ensure stable database connectivity
Log Warnings and Errors	Count of warning/error logs generated per hour	< 10 warnings/hour	Identify emerging issues proactively

2. Monitoring Tools and Alerting Logic

Tools:

- **Grafana** – Dashboard visualization and threshold monitoring
- **Datadog** – Application performance monitoring (APM) and error tracking
- **AWS CloudWatch** – Infrastructure and resource monitoring
- **Slack Integration** – Real-time alert notifications

Alerting Logic:

- Alerts are automatically triggered when any metric exceeds its threshold for **5 consecutive minutes**
- Alerts are routed to the **#devops-alerts** Slack channel and to **PagerDuty** for high-severity issues
- Alert details include timestamp, affected component, severity, and recommended first steps

Escalation Protocol:

Severity	Description	First Responder	Escalation Path
High	System outage, payment failure, or database down	DevOps (SRE Lead)	Engineering Manager → CTO

Severity	Description	First Responder	Escalation Path
Medium	Elevated API latency, intermittent errors	DevOps Engineer	Application Developer → SRE Lead
Low	Log warnings, minor anomalies	Support/QA Engineer	DevOps Engineer (if persistent)

Section 2: Maintenance Checklist

Daily Tasks

- Review application and system logs for errors or warnings
- Monitor system uptime, CPU, and memory usage dashboards
- Confirm successful completion of automated backups
- Check payment gateway and order processing services
- Verify monitoring alerts are functional and acknowledged

Weekly Tasks

- Review database size and growth trends
- Analyze alert history to identify recurring issues
- Test restoration of a random backup
- Validate SSL certificates and API keys' validity
- Conduct minor performance tuning (index optimization, cache refresh)

Monthly Tasks

- Apply OS and application-level security patches
- Review user activity and access permissions
- Analyze long-term performance and usage trends
- Archive old logs and rotate storage files
- Conduct a failover and disaster recovery drill

Incident Response and Escalation Workflow

Severity Levels:

- **High:** System unavailability, data loss, or financial transaction failure
- **Medium:** Partial service degradation or slow performance
- **Low:** Non-critical warnings or cosmetic errors

Response Steps:

1. **Detection:** Monitoring tools trigger an alert
2. **Notification:** Alert sent via Slack and PagerDuty to the on-call engineer
3. **Investigation:** First responder reviews system logs, dashboards, and metrics
4. **Resolution:** Issue fixed or mitigated; temporary workarounds applied if needed
5. **Communication:** Status updates shared via Slack and Jira/ServiceNow incident ticket
6. **Post-Mortem:** Within 48 hours, document root cause, resolution steps, and preventive actions

Continuous Improvement

- Review monitoring effectiveness after each incident
- Update alert thresholds based on observed system behavior
- Regularly assess new tools or integrations to enhance monitoring accuracy