

Formation Tests de charge et de performance : méthodologie et panorama des outils

Réf. SF33466

Acquérez une vision complète de la démarche de test de charge, du cadrage au diagnostic, pour garantir la performance de vos applications, quel que soit l'outil.

2 j Durée	1 500 € HT Inter / participant	3 700 € HT Intra (≤12 pers.)	0 Sessions à venir
--------------	-----------------------------------	---------------------------------	-----------------------

Objectifs pédagogiques

- ✓ Distinguer les différents types de tests de charge et de performance (charge, stress, endurance, pic, robustesse, capacity planning) et situer leur place dans le cycle de vie applicatif
- ✓ Identifier les indicateurs de performance clés (temps de réponse et percentiles, débit, taux d'erreur) et les ressources à surveiller (CPU, mémoire, réseau, moniteurs serveurs)
- ✓ Construire un plan de charge réaliste à partir d'un usage métier (profils utilisateurs, scénarios, volumétrie)
- ✓ Interpréter les résultats d'une campagne pour diagnostiquer les goulots d'étranglement et formuler des préconisations dans un rapport de synthèse
- ✓ Gérer la phase de Comparer les principaux outils du marché (JMeter, Gatling, k6, Locust) et choisir l'outillage adapté à un contexte donné

Public visé & prérequis

Public visé Chefs de projet, Développeurs, Ingénieurs DevOps, Testeurs logiciels

Prérequis- Connaître le fonctionnement d'une application web n-tiers (client/serveur, HTTP, base de données), tous langages confondus - Disposer de notions de base sur les indicateurs système (CPU, mémoire, réseau)

Programme détaillé

Cadrage des tests de charge et de performance

- Performance et charge : distinguer la rapidité d'une application de son comportement sous sollicitation croissante
- Place du test dans le cycle de vie applicatif : tester au plus tôt plutôt que subir en production
- Typologie des tests : charge, stress (limites), endurance (soak), pic (spike), robustesse, capacity planning, non-régression de performance
- Enjeux et objectifs : disponibilité, temps de réponse, SLA, KPI et gestion des risques
- Système n-tiers et sources de dysfonctionnement : où naissent les problèmes de performance (front, applicatif, base de données, réseau)
- Coûts, gains et ROI d'une démarche de test : arbitrer l'effort selon la criticité de l'application

Activités pratiques :

- Qualification d'un besoin de test : choisir le(s) type(s) de test adapté(s) à différents contextes (lancement d'un nouveau service, pic saisonnier, migration d'infrastructure)
- Atelier de cadrage : définir les objectifs et le périmètre d'une campagne pour une application web de référence

Métrologie de la performance

- Indicateurs côté utilisateur : temps de réponse, percentiles (P90/P95/P99), débit (throughput), taux d'erreur, niveau de concurrence
- Raisonner en percentiles : limites de la moyenne et lecture de la distribution des temps de réponse
- Ressources système à surveiller : CPU, mémoire, I/O disque, réseau et comportement du garbage collector
- Moniteurs et points de mesure : serveurs web, serveurs d'application et bases de données
- Observabilité et APM (notion) : situer les solutions modernes (Datadog, Dynatrace, New Relic, Grafana/Prometheus) dans une démarche de test
- Représentativité des mesures : conformer la plateforme de test à la plateforme de production

Activités pratiques :

- Définition d'un tableau de bord d'indicateurs pour une application cible (métriques utilisateur + ressources)
- Lecture commentée d'un relevé de métriques système pendant une montée en charge

Méthodologie d'une campagne de test

- Étude de faisabilité et modèle de charge : traduire un usage métier en profils d'utilisateurs et volumétrie
- Construction des scénarios : parcours utilisateurs représentatifs, jeux de données et montée en charge progressive
- Scripting, corrélation et paramétrage (principes) : rejouer des scénarios réalistes et gérer les données dynamiques (sessions, jetons)
- Stratégie de tir : tirs préliminaires, tirs nominaux et conditions de répétabilité
- Préparation des ressources et prérequis : environnement, données de test et planification
- Pilotage de l'exécution : injection de charge, surveillance en temps réel et collecte des données

Activités pratiques :

- Construction d'un plan de charge réaliste à partir d'un usage métier (parcours d'achat en ligne, consultation de comptes, soumission de formulaires)
- Conception de scénarios et identification des points de corrélation/paramétrage sur une application de référence

Analyse, diagnostic et restitution

- Lecture des résultats : courbes de temps de réponse, débit et erreurs en fonction de la charge
- Corrélation charge / ressources : relier la dégradation observée à la consommation système
- Identification des goulots d'étranglement : front, applicatif, base de données, réseau et configuration
- Profilage applicatif (notion) : descendre au niveau du code pour localiser un point chaud
- Préconisations d'amélioration : pistes d'optimisation (architecture, code, réglage serveur, mise à l'échelle)
- Rapport de campagne : indicateurs, constats, risques et préconisations dans un livrable exploitable par les décideurs

Activités pratiques :

- Interprétation d'un jeu de résultats de campagne : formuler un diagnostic et des préconisations
- Rédaction d'une trame de rapport de campagne synthétique

Panorama des outils et choix de l'outillage

- Familles d'outils : pilotage par protocole ou par navigateur, open source ou commercial, on-premise ou cloud
- JMeter : référence open source historique, approche par protocole et large couverture
- Gatling : approche « as code », scénarios expressifs et rapports détaillés
- k6 : test de charge « as code » orienté développeurs et SRE, scripting en JavaScript
- Locust : test de charge en Python avec modèle d'injection distribué
- Critères de choix d'un outil : protocole cible, compétences de l'équipe, intégration, reporting et coût
- Injection de charge distribuée et cloud (notion) : générer de la charge à grande échelle

Activités pratiques :

- Démonstration comparée de plusieurs outils (JMeter, Gatling, k6, Locust) sur la même application web
- Élaboration d'une grille de choix d'outil adaptée à un contexte projet donné

Pourquoi choisir Sparks Formation ?

20 ans d'expertise IT, des formateurs experts et des sessions garanties contre l'annulation.

20 ans
d'expertise

1 068+
formations

11 178
apprenants formés/an

+67
Net Promoter Score

Avis des participants

Note moyenne des participants de la filière Qualité Logicielle & Testing (QA) : 4,9/5 — 11 avis

Patrick F. — Merci à Gianni pour cette formation

Remi C. — Un formateur passionné, très investi dans la transmission de son savoir, rendant la formation claire et enrichissante.

Votre formateur

Dhafrallah D. — Formateur

Dhafrallah intervient notamment sur les langages et écosystèmes backend, ainsi que sur le développement web et fullstack. La conception de supports pédagogiques fait également partie de ses activités.

Ils nous font confiance

Cette année notre filière Qualité Logicielle & Testing (QA) a accompagné plus de 31 entreprises dont :

ORANGE BUSINESS SERVICES SA SERCY ADEO SERVICES ADSN ADVENTIEL APSIDE TOP APSIDE TOULOUSE BANDAI NAMCO ENTERTAINMENT EUROPE CEA GRENOBLE CEGID +21 autres

Financement & certifications

Cette formation est éligible aux financements OPCO et FNE-Formation. Nos équipes vous accompagnent dans le montage de votre dossier.



Réservez votre place dès maintenant

20 ans d'expertise · 1 000+ formations · NPS +67 · réponse sous 24h ouvrées

0 805 950 800

demande@sparks-formation.com