Corrigé : Évaluation Architectures et Hébergement

1️⃣ QCM (Réponses)

1. N-tiers
2. Base de données
3. Docker
4. Distribuer les requêtes vers plusieurs serveurs backend
5. Virtualisation avec VPS

2️⃣ Vrai/Faux (Réponses et justifications)

6. Vrai. Une API REST permet de structurer les échanges entre modules, même dans un monolithe.
7. Faux. Un hébergement mutualisé partage ses ressources entre plusieurs clients, ce qui peut limiter les performances.
8. Faux. Docker isole uniquement les processus applicatifs tandis qu'une VM inclut un OS complet.
9. Vrai. Chaque service étant indépendant, il peut être mis à jour ou scalé sans impacter les autres.
10. Faux. Les sauvegardes protègent contre la perte de données due à des erreurs humaines ou à une panne matérielle.

3️⃣ Questions courtes (Réponses attendues)

11. Une architecture n-tiers sépare les couches (frontend, backend, base de données), limitant ainsi les risques d'attaques.
12. Un VPS offre des ressources dédiées et plus de contrôle, tandis qu'un hébergement mutualisé partage les ressources et réduit les performances.
13. Un administrateur peut utiliser un pare-feu, limiter l'accès SSH et appliquer le principe du moindre privilège.
14. Les logs permettent de surveiller l'activité, détecter les erreurs ou tentatives d'intrusion et analyser la performance.
15. Pour une application simple, les microservices ajoutent une complexité inutile en multipliant les points de communication.

4️⃣ Étude de cas (Réponses détaillées)

16. Recommandation d'une architecture n-tiers.
17. Hébergement sur un VPS avec une possible évolution vers le cloud.
18. Mise en place d'un load balancing, optimisation SQL et caching.
19. Sécurisation des accès, protection contre les attaques et sauvegardes régulières.

5️⃣ Rédaction technique (Réponses détaillées)

20. Étapes d'installation d'un serveur VPS avec configuration et sécurisation.
21. Stratégies pour sécuriser une API REST (authentification, validation des entrées, chiffrement).
22. Optimisation des performances backend (cache, optimisation des requêtes, scalabilité).

Total de l'évaluation : 60 points.