**Plan d'Action de Test**

**1. Introduction**

* **Objectif** : Assurer la qualité et la fiabilité de l'application web.
* **Scope** : Tests du frontend (Next.js) et de l'API (Node.js, Express, Mongoose, Redis).

**2. Types de Tests**

**2.1. Tests Unitaires**

* **Frontend** : Utiliser Jest et React Testing Library pour tester les composants React.
* **API** : Utiliser Mocha/Chai ou Jest pour tester les routes et les contrôleurs de l'API.

**2.2. Tests d'Intégration**

* Vérifier l'interaction entre les composants du frontend et les API.
* Tester les appels API dans des scénarios simulés.

**2.3. Tests End-to-End (E2E)**

* Utiliser Cypress ou Playwright pour simuler des scénarios utilisateur complets.
* Tester le flux de l'utilisateur depuis le chargement de la page jusqu'à l'interaction avec l'API.

**2.4. Tests de Performance**

* Utiliser des outils comme JMeter ou k6 pour tester la performance sous charge.
* Évaluer les temps de réponse des API et la capacité à gérer plusieurs connexions simultanées.

**2.5. Tests de Sécurité**

* Effectuer des tests de vulnérabilité sur l'API pour détecter les failles potentielles (injections SQL, XSS, CSRF).
* Utiliser des outils comme OWASP ZAP pour automatiser certaines vérifications.

**3. Environnement de Test**

* **Frontend** : Développer dans un environnement de test isolé.
* **API** : Utiliser des bases de données de développement et de test (MongoDB et Redis).
* **Automatisation** : Configurer CI/CD pour exécuter les tests automatiquement lors des déploiements.

**4. Outils de Test**

* **Frontend** : Jest, React Testing Library, Cypress/Playwright.
* **API** : Mocha/Chai ou Jest, Postman pour tester les endpoints manuellement.
* **Performance** : JMeter, k6.
* **Sécurité** : OWASP ZAP, Snyk.

**5. Scénarios de Test**

* **Frontend** :
  + Vérifier le rendu des composants.
  + Tester les interactions utilisateur (clics, soumissions de formulaires).
  + Vérifier la gestion des états et des erreurs.
* **API** :
  + Tester les endpoints pour les méthodes GET, POST, PUT, DELETE.
  + Vérifier les réponses pour différents scénarios (succès, erreurs, validations).
  + Tester l'intégration avec MongoDB et Redis pour s'assurer que les données sont correctement stockées et récupérées.

**6. Rapport et Suivi**

* Documenter les résultats des tests et les anomalies détectées.
* Mettre en place un tableau de bord pour suivre l'état des tests et des corrections.
* Planifier des revues régulières pour ajuster les stratégies de test.

**7. Conclusion**

* Évaluer les résultats des tests pour déterminer la préparation à la mise en production.
* Réviser le plan de test périodiquement en fonction des nouvelles fonctionnalités et des retours utilisateurs.