

Rapport de Tests

INFO3 – Base de données NoSQL

Gestion de Restaurant

Groupe : F
Auteurs : OUBERKA Mohamed
BOUCHELLAL Imad-Eddine
OUKACI Djedjiga
ADRAR Céline
Encadré par : Eddy Kiomba



Document Version : 1.0
Submission date : 21-11-2025

1 Tests API avec `test.http`

Le fichier `test.http` permet de tester l'ensemble des fonctionnalités du backend de l'application de gestion de restaurant (création d'utilisateurs, plats, commandes, mise à jour des statuts et statistiques) en utilisant l'extension *REST Client* de Visual Studio Code.

1.1 Prérequis

- Backend démarré sur `http://localhost:5000`.
- Visual Studio Code installé.
- Extension *REST Client* installée.
- Fichier `test.http` ouvert dans VS Code.

1.2 Étapes générales pour exécuter les tests

1. Ouvrir le dossier `Backend` dans VS Code.
2. Démarrer le serveur avec `npm start` ou `npx nodemon server.js`.
3. Ouvrir le fichier `test.http`.
4. Pour chaque requête, cliquer sur le lien *Send Request* (ou utiliser le raccourci clavier) situé au-dessus de la ligne `GET/POST/PUT`.
5. Copier les identifiants (`_id`) renvoyés par certaines réponses (utilisateurs, plats, commandes) et les remplacer dans les requêtes suivantes aux endroits indiqués (`Server1Id`, `Plat1Id`, `Plat2Id`, `Commande1Id`, etc.).

1.3 Phase 1 : Création des données (F1 à F5)

1. F5 – Créer un utilisateur gérant

Envoyer la requête POST `/api/utilisateurs` avec le corps JSON contenant un nom, le rôle « Gérant » et un mot de passe. Noter l'identifiant retourné (AdminGérantId).

2. F5 – Créer un serveur

Envoyer une seconde requête POST `/api/utilisateurs` avec le rôle « Serveur ». Noter l'identifiant retourné (Serveur1Id).

3. F1 – Créer un premier plat

Envoyer POST `/api/plats` avec un plat de type « Entrée » (par exemple « Soupe »). Noter l'identifiant du plat (Plat1Id).

4. F1 – Créer un deuxième plat

Envoyer POST `/api/plats` pour un plat de type « Plat » (par exemple « Filet »). Noter l'identifiant du plat (Plat2Id).

5. F2 – Créer une commande

Envoyer POST `/api/commandes` avec un corps JSON contenant :

- `numeroTable` (par exemple 12),
- `serveurId` (remplacer `Server1Id` par l'ID réel du serveur),
- un tableau `items` avec les `platId` (Plat1Id, Plat2Id) et leurs quantités.

Noter l'identifiant de la commande (Commande1Id).

1.4 Phase 1 bis : Cycle de vie de la commande (F3, F4)

1. F3 – Marquer la commande comme « Servie »

Envoyer PUT `/api/commandes/Commande1Id/statut` avec le corps JSON `"statut": "Servie"` en remplaçant `Commande1Id` par l'ID réel.

2. F4 – Marquer la commande comme « Payée »

Envoyer de nouveau PUT `/api/commandes/Commande1Id/statut` avec `"statut": "Payée"`. Cette étape clôture la transaction et enregistre les informations nécessaires pour les statistiques (chiffre d'affaires, temps de service, etc.).

1.5 Phase 2 : Statistiques (F6 à F10)

Après avoir créé au moins une commande payée, exécuter les requêtes suivantes :

1. **F6 – Chiffre d'affaires total**

Envoyer GET `/api/stats/chiffre-affaires` pour calculer le chiffre d'affaires cumulé à partir des commandes payées.

2. **F7 – Plats les plus populaires**

Envoyer GET `/api/stats/popular-plats` pour obtenir la liste des plats les plus vendus.

3. **F8 – Temps de service moyen**

Envoyer GET `/api/stats/avg-service-time` pour calculer le temps moyen entre la création, le service et le paiement des commandes.

4. **F9 – Performance des serveurs**

Envoyer GET `/api/stats/server-performance` pour obtenir, par serveur, le chiffre d'affaires généré et éventuellement d'autres indicateurs de performance.

5. **F10 – Ventes par catégorie**

Envoyer GET `/api/stats/sales-by-category` pour agréger les ventes par catégorie de plat (Entrée, Plat, Dessert, etc.).

1.6 Interprétation des résultats

Les réponses JSON renvoyées par ces appels permettent de vérifier :

- la bonne création et mise à jour des ressources (utilisateurs, plats, commandes),
- la cohérence du cycle de vie d'une commande (« Ouverte », « Servie », « Payée »),
- la validité des agrégations MongoDB pour le chiffre d'affaires, les plats populaires, le temps de service moyen, la performance des serveurs et les ventes par catégorie.