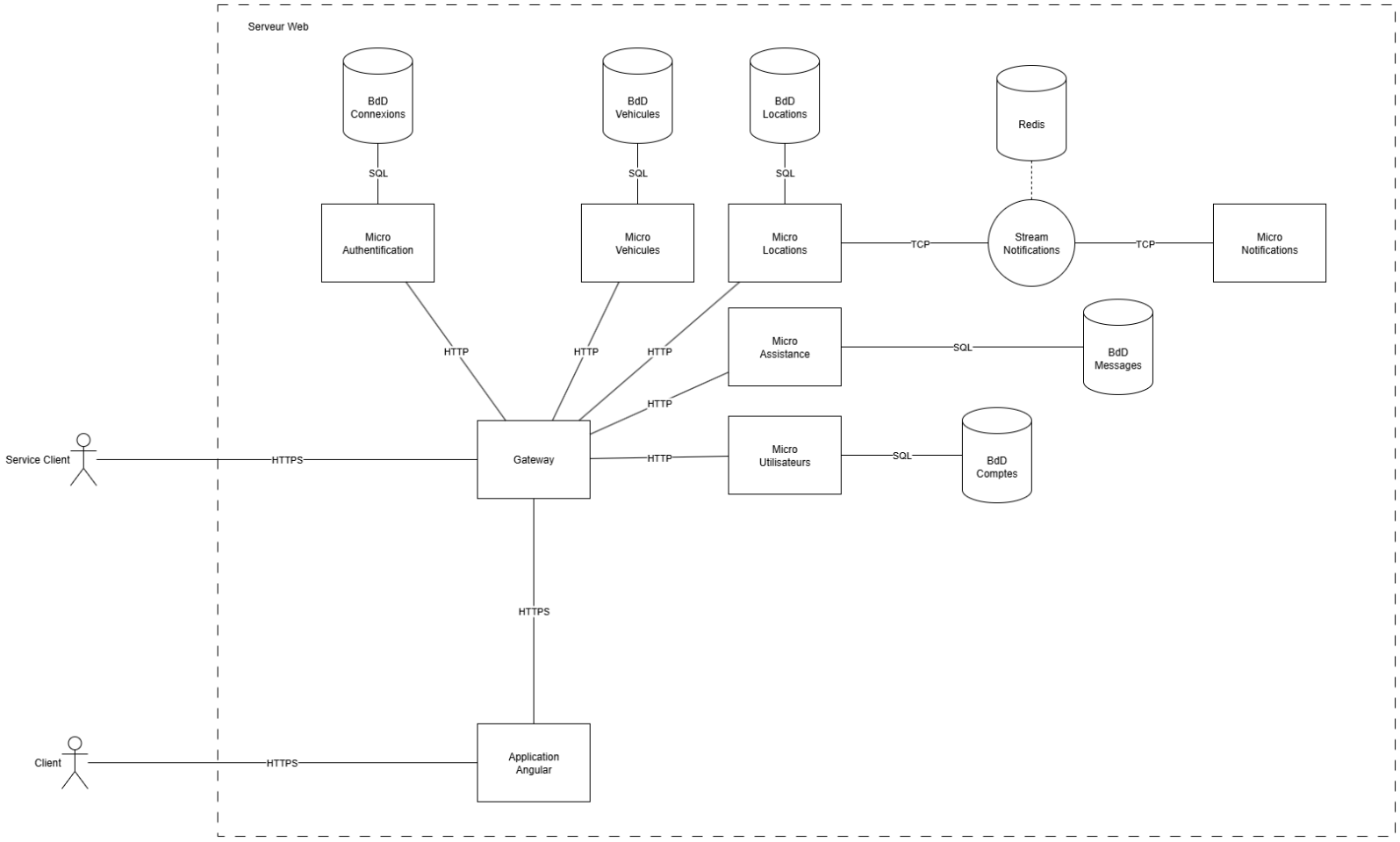


**.. Description générale de l’architecture**



**.. Liste de contrôle de l’architecture terminée**

# Composants logiciels

* Suivi du cycle de vie de l’applicatif via l’utilisation de version control
  + Utilisation de GitHub pour le stockage de dépôts et la mise en place du projet
  + Réalisation de pipelines CI / CD pour automatiser l’ensemble des démarches de version control
* Réalisation de test unitaires pour le panel des microservices
  + Utilisation de JUnit / Mockito
  + Objectif d’avoir au minimum 80% de couverture du code
  + Utilisation de TestContainers pour la réalisation des tests de sorte à ne pas intervenir au niveau des bases de données de production
* Réalisation de tests unitaires pour l’application frontend
  + Utilisation de Jest
  + Objectif d’un score de couverture du code d’au moins 80%
* Tests de type E2E
  + Utilisation de Cypress
  + Réalisation de stubs pour éviter de perturber l’intégrité des données de notre applicatif
* Mise en place de revues du code obligatoire dans le cadre du développement
  + A la suite d’un pipeline de type CI
  + Réalisée par un développeur qualifié de type lead technique
* Réalisation de scan de vulnérabilités des applicatifs
  + Utilisation de Sonarqube dans le but d’obtenir un score de qualité du code
  + Passage des applicatif via Trivy pour obtenir un rapport des vulnérabilités logicielles présentes dans la CVE

# Services ou composants tiers logiciels

* Vérification des licences des applications utilisées
  + MIT, OpenSource, Apache, etc… Eviter les licences pouvant poser problèmes lors de l’intégration de ces services externes à notre applicatif
* Vérification de la conformité à la RGPD pour ces services externes avant leur intégration à notre développement personnel
* Analyse de potentiels contrats / SLA dans le but d’obtenir des garanties vis-à-vis de la disponibilité de l’applicatif / du support client en cas de nécessité de les contacter
* Scan des vulnérabilités des images Docker tierces
* Suivi via une veille technologique des évolutions futures des applicatifs (patchs, correctifs, etc.)

# Gestion des données

* Localisation des données sur le territoire français pour le respect de la RGPD (Azure France dans le cadre d’un déploiement en nuage, voire réalisation d’un déploiement en cloud privé via OpenStack)
* Classification des données de sorte à identifier ce qui est sensible ou non.
* Chiffrement des données en repos (via par exemple Azure Storage Encryption)
* Chiffrement des données en transit par l’usage de TLS et donc de l’https lors de la communication des informations entre les couches applicatives
* Anonymisation des logs et des données de métriques
* Mise en place d’une politique de rétention et de suppression du compte en cas de volonté du client
* Réalisation d’un backup journalier de la base de données (aux alentours de 2h00) et stockage de ce backup via l’utilisation de la règles 3-2-1

# Infrastructure

* Panel de microservices :
  + Contrôle de la santé via l’utilisation de LivenessProbe et de ReadinessProbes intégrées à Kubernetes
  + Contrôle de la charge et du statut des microservices via l’utilisation de Prometheus et de Grafana dans le but de pouvoir suivre la santé en un endroit commun
* Application frontend :
  + Contrôle de la santé via l’utilisation de LivenessProbe / ReadinessProbe intégrées à Kubernetes
  + Utilisation des endpoints /health disponible pour Prometheus dans le cadre de la récupération des métriques
* Centralisation des logs des différents applicatifs
  + Utilisation d’Elastic Stack (Elasticsearch) pour accélérer la réalisation des requêtes de récupération des logs à l’avenir
* Politique de redémarrage automatique des pods en cas de soucis
  + Self-healing policies
* Réalisation de tests de montée en charge afin de pouvoir anticiper les moments d’affluence
* Gestion de secrets via l’utilisation d’un environnement sécurisé
  + Azure Key Vault / Kubernetes Secrets

# Sécurité

* Utilisation d’API Stateless et mode d’authentification sécurisé
  + JWT obligatoire transféré via l’utilisation du mode http-only et des headers de la requête
* Mise en place d’une politique d’authentification renforcée
  + Utilisation de Gaptcha lors de l’enregistrement
  + Possibilité d’ajouter une authentification à plusieurs facteurs via le scan d’un code QR par le client
* Sécurisation via le principe du less privilege
  + Réalisation d’un ensemble de policies dans le but de procéder à la mise en place d’une PBAC
* Sécurisation de la Gateway via l’utilisation du protocole HTTPS et de certificats TLS récents (mise à jour automatique de ces derniers via par exemple certbot et let’s encrypt)
* Mise en place d’une protection contre les attaques les plus risquées
  + Consultation du OWASP Top Ten durant la veille technologique
  + Mise en place de protection contre le brute force, les DoS, le XSS, etc…
* Réalisation d’audit des logs dans le but de déceler des comportements douteux ou le passage de données sensibles
* Réalisation de tests d’intrusion de façon régulière afin de vérifier la conformité des applicatifs
* Conformisation des applicatifs à la RGPD (droit à l’oubli, demande de consentement explicite et minimisation des logs)

