



PLATE-FORME DIGITALE

CAHIER DES CHARGES

Exprimez clairement le besoin et le périmètre du projet



CAHIER DES CHARGES

Nom / Code projet	AgriLink
-------------------	----------

Référence	
-----------	--

Chef de projet	Baraka Kamal Madiafi Mohammed
----------------	----------------------------------

Service/Organisation	Groupe 4ème Année Info
----------------------	------------------------

Historique			
Version	Auteur	Description	Date
001	Baraka Kamal	Véersion initiale	01/03/2025
002	GIIA4	V2	09/03/2025
003	GIIA4	V3	18/03/2025

Contributions à la V3 du cahier				
Groupe / Personne	Partie rajoutée	Partie modifiée	Description	Date
Alban NYANTUDRE	<ul style="list-style-type: none"> - Plateforme WaziUp et son API 	<ul style="list-style-type: none"> - Contexte, - Objectifs, - périmètre, - aspects fonctionnels - aspects techniques 	Rendu le cahier plus clair et spécifique en détaillant chaque partie. Rajout de comment la plateforme va interagir avec le logiciel cloud WaziUp.	18/03/2025
Groupe 8	<ul style="list-style-type: none"> - Budget - Délais détaillés - Clauses contractuelles - Spécifications techniques complémentaires - Annexes 	<ul style="list-style-type: none"> - Budget jusqu'à la fin 	Détail sur les aspects financiers, délais détaillés et des spécifications techniques complémentaires	18/03/2025

Table des matières

Contexte du projet	4
Objectifs du projet	4
Périmètre du projet	4
Aspects fonctionnels	5
Aspects techniques	6
Ressources	7
Délais	8
Budget	8

Contexte du projet

Contexte du projet

Le projet AgriLink vise à concevoir une **plateforme digitale dédiée à la collecte, au stockage, au traitement et à la visualisation de données issues de capteurs IoT** (Internet of Things) intelligents déployés dans des environnements agricoles. Ces capteurs, installés sur différents sites et connectés via des réseaux sans fil (LoRaWAN, Wi-Fi), transmettent leurs données à des passerelles locales, qui les acheminent vers une infrastructure cloud pour un traitement ultérieur. L'objectif est d'installer plusieurs passerelles qui, une fois connectées à la plate-forme centrale, permettront une gestion flexible et évolutive des données collectées. La plate-forme ne se contente pas d'afficher les résultats en temps réel, elle assure également le stockage des données pour une consultation ultérieure et un traitement avancé (analyse, visualisation des tendances, alertes, etc.).

Pour éviter de dupliquer les efforts existants, la plateforme s'appuie sur **Waziup, une solution open source spécialisée dans l'IoT, qui fournit déjà une infrastructure complète incluant des API de gestion des capteurs, des passerelles et des données**. Ainsi, l'innovation principale réside dans l'exploitation de l'API Waziup pour orchestrer le flux de données, tandis que le développement se concentre sur la **couche applicative** :

- Extraction des données brutes via les endpoints de Waziup API, transformation (nettoyage, enrichissement), et chargement vers une base de données dédiée.
- Utilisation d'une base de données / stockage scalable pour l'historique des données, complémentaire au stockage temporaire de Waziup.
- Création d'une interface utilisateur sur mesure pour les agriculteurs et chercheurs, connectée en "temps réel".

Ce processus vise à optimiser la prise de décision en fournissant des informations précises et exploitables à partir des données brutes capturées par les capteurs. La plateforme AgriLink capitalise ainsi sur la robustesse de Waziup pour la gestion des devices IoT, tout en ajoutant une valeur métier via des outils d'analyse et de décision adaptés à l'agriculture de précision.

Objectifs du projet

Objectifs du projet

L'objectif principal de ce projet est de développer une plateforme digitale capable de répondre aux besoins spécifiques de deux catégories d'utilisateurs finaux, à savoir les agriculteurs et les chercheurs, tout en exploitant pleinement les atouts de la solution Waziup. Pour garantir la robustesse et la sécurité de l'ensemble du système, l'intervention d'un administrateur principal s'avère cependant indispensable.



1. Administrateur

L'administrateur joue un rôle central dans la gestion et la centralisation des dispositifs IoT. Il est responsable de la **configuration initiale et de l'intégration des capteurs et des passerelles via le dashboard Waziup** (accessible sur <https://dashboard.waziup.io/>), où il gère notamment la création et le renouvellement des tokens d'accès afin d'assurer la sécurité des échanges. En parallèle, il surveille en continu l'**état des capteurs et des passerelles, détecte les dysfonctionnements ou défaillances matérielles** et met en place des alertes critiques pour maintenir la disponibilité optimale du système. Ce pilotage centralisé permet d'harmoniser la collecte des données provenant de divers dispositifs et de garantir une gestion évolutive et fiable sur l'ensemble du cloud Waziup.

2. Les agriculteurs

Pour les agriculteurs, la plate-forme vise à offrir un **affichage en temps réel des données collectées par les capteurs**. L'objectif est de fournir une aide à la décision simple et efficace, notamment pour optimiser l'irrigation et la gestion des ressources en eau. Grâce à cette solution, les agriculteurs pourront mieux gérer leur consommation d'eau et d'autres ressources, améliorant ainsi leur efficacité et réduisant les coûts liés à l'irrigation.

3. Les chercheurs

Pour les chercheurs, l'objectif est de fournir une plate-forme plus avancée permettant non seulement l'affichage en temps réel des données, mais aussi leur **traitement et leur analyse approfondie**. Les chercheurs pourront ainsi utiliser la plate-forme pour effectuer des **analyses de données, prédire des tendances et prendre des décisions éclairées** basées sur les informations collectées. Ce traitement de données avancé offrira des outils pour des études à long terme, ainsi que des modèles prédictifs adaptés à leurs besoins de recherche.

Objectif transversal : Une architecture unifiée

La plateforme AgriLink tire parti de l'API Waziup pour interconnecter capteurs, passerelles et utilisateurs à travers le monde dans un écosystème cohérent. Elle garantit :

- **Interopérabilité:** compatibilité avec les protocoles IoT standards (LoRaWAN, MQTT) via Waziup.
- **Centralisation des données:** stockage unifié des historiques des données provenant de diverses localisations géographiques dans une base de données (comme MongoDB Atlas), synchronisé avec le cloud Waziup.
- **Évolutivité technique / scalabilité:** capacité à intégrer de nouveaux capteurs ou services sans refonte majeure, grâce à l'approche modulaire de Waziup.



Périmètre du projet

Périmètre du projet

Le présent projet se concentre **exclusivement sur le développement d'une plateforme web** destinée à la gestion, au stockage et au traitement des données recueillies par des capteurs IoT.

En tant que chef de projet, ayant déjà assuré l'installation et la mise en place de la partie IoT (capteurs et passerelles), le champ d'intervention se limite désormais à la conception, à la réalisation et à la mise en service de la plateforme. Cette dernière doit permettre de centraliser les informations issues des dispositifs IoT et d'offrir une interface conviviale pour la visualisation en temps réel ainsi que pour l'analyse avancée des données.

1. Ce qui est inclus dans le projet:

Le périmètre du projet couvre la conception, le développement et le déploiement d'une plateforme digitale centralisée pour la gestion des données IoT agricoles.

Les activités incluses sont structurées autour de cinq axes majeurs :

• **Interface utilisateur (UI/UX) :** Une application web moderne et réactive sera développée pour répondre aux besoins distincts des agriculteurs et des chercheurs.

- Les agriculteurs bénéficieront d'un tableau de bord simplifié avec des visualisations en temps réel (graphiques linéaires, cartes interactives) et des alertes automatisées.
- Les chercheurs auront accès à une interface analytique avancée, incluant des outils de modélisation statistique, des prédictions basées sur des algorithmes de machine learning, et des fonctionnalités d'export de données.

La personnalisation des vues (filtres par capteur, période, type de donnée) sera intégrée pour améliorer l'expérience utilisateur.

• **Stockage des données :** Un système de stockage robuste et évolutif sera mis en place, combinant une base de données cloud NoSQL pour l'historique à long terme et un cache temporaire pour optimiser l'accès aux données fréquemment consultées.

La sécurité des données sera garantie par des protocoles de chiffrement et des sauvegardes automatisées.

• **Traitement des Données (ETL) :**

Des pipelines de traitement seront développés pour extraire les données brutes depuis la plateforme Waziup, les nettoyer (élimination des valeurs aberrantes), les enrichir (ajout de métadonnées géographiques) et les charger dans la base de données principale.

Ces processus permettront de transformer les données brutes en informations exploitables pour les analyses.

- **Intégration des passerelles IoT :**

La plateforme AgriLink se connectera aux passerelles IoT existantes **via l'infrastructure Waziup**, en utilisant ses capacités natives de gestion des protocoles de communication (LoRaWAN, MQTT). Cette intégration permettra une centralisation transparente des données sans nécessiter de modifications matérielles.

- **Formation des utilisateurs :**

Des supports pédagogiques multiformats (tutoriels / articles de blog intégrés, guides PDF) seront élaborés.

Les agriculteurs seront formés à l'utilisation du tableau de bord, tandis que les chercheurs apprendront à exploiter les outils d'analyse avancée.

2. Ce qui est exclu du projet :

Certaines activités sont explicitement exclues du périmètre du projet.

- **La gestion et la maintenance des capteurs et passerelles IoT :**

La configuration matérielle, la calibration des capteurs et la maintenance des passerelles relèvent de la responsabilité de Waziup. Cette partie étant déjà réalisée, n'entre pas dans le cadre du projet actuel.

- **L'infrastructure physique des capteurs et passerelles :**

L'installation physique des capteurs et passerelles, ainsi que leur maintenance à long terme, sont exclues du périmètre.

En d'autres termes, aucune intervention sur l'installation des antennes LoRa, des réseaux de communication ou des capteurs sur le terrain.

- **Le développement d'API personnalisées :**

La plateforme AgriLink s'appuie exclusivement sur les fonctionnalités existantes de Waziup, sans création de nouvelles API.

- **L'éventuelle extension géographique :**

Le projet se concentrera sur la connectivité des passerelles et des capteurs à la plate-forme, sans gestion immédiate de zones géographiques étendues.

3. Taches principales:

- **Chef de projet :** Gestion globale du projet, supervision de l'intégration IoT, coordination avec les étudiants pour le développement de la plate-forme.

- **8 groupes de maîtres ingénieurs en informatique :** La conception et le développement de la plate-forme seront répartis entre 8 groupes d'étudiants.



Chaque groupe aura des responsabilités spécifiques, telles que :

- **Tâche 1 (UI/UX):** Conception et développement de l'interface utilisateur.
- **Tâche 2 (Stockage):** Configuration de la base de données cloud et optimisation des performances de stockage.
- **Tâche 3 (ETL):** Implémentation des pipelines ETL pour le nettoyage et l'enrichissement des données.
- **Tâche 4 (Analyse):** Développement des outils d'analyse statistique et des modèles prédictifs.
- **Tâche 5 (IoT):** Intégration des passerelles IoT et gestion des connexions (**À REMPLACER**)
- **Tâche 6 (Sécurité):** Mise en œuvre de l'authentification, chiffrement des données et gestion des rôles utilisateurs.
- **Tâche 7 (Alertes):** Configuration du système de notifications et personnalisation des seuils d'alerte.
- **Tâche 8 (Tests):** Exécution de tests de charge, validation des fonctionnalités et documentation des bugs.

4. Contraintes liées à Waziup

- **Quotas d'appels API :** Waziup impose des limites strictes sur le nombre de requêtes autorisées par minute, nécessitant une optimisation des appels via des stratégies de mise en cache.
- **Compatibilité des données :** Les capteurs doivent être préalablement enregistrés dans Waziup avec des formats de données standardisés pour assurer leur interopérabilité avec AgriLink.

Aspects fonctionnels

Description fonctionnelle

La plate-forme digitale sera conçue pour répondre aux besoins spécifiques des agriculteurs et des chercheurs. Elle offrira une interface utilisateur adaptée à chaque profil, tout en garantissant un accès rapide et fiable aux données collectées par les capteurs IoT.

1. Ingestion des Données

La plateforme AgriLink intègre un mécanisme robuste pour collecter les données générées par les capteurs IoT via l'infrastructure Waziup. Ce processus comprend :

- **Collecte en temps réel :** Les données sont récupérées périodiquement (toutes les 5 minutes par défaut) à partir des capteurs enregistrés dans Waziup, en utilisant des appels



API programmés. Un système de vérification garantit la complétude des données (ex : détection des paquets manquants).

• **Formats supportés** : Les données brutes sont ingérées dans des formats standardisés (JSON, CSV) alignés sur les modèles de Waziup, avec des métadonnées incluant l’ID du capteur, l’horodatage, et la localisation géographique.

2. Transformation des Données

Un pipeline ETL (Extract-Transform-Load) est implémenté pour convertir les données brutes en informations exploitables :

• **Nettoyage** :

- Suppression des valeurs aberrantes (ex : humidité > 100%, températures irréalistes pour une zone géographique).
- Imputation des valeurs manquantes via des méthodes statistiques (moyenne mobile, interpolation linéaire).

• **Enrichissement** :

- Ajout de métadonnées contextuelles (ex : nom de l’exploitation agricole, type de culture associé au capteur).
- Normalisation des unités de mesure (ex : conversion des °F en °C pour l’internationalisation).

• **Structuration** :

- Agrégation des données par intervalles temporels (ex : moyennes horaires pour réduire la volumétrie).
- Formatage des timestamps selon le standard ISO 8601 pour assurer la cohérence.

3. Stockage et gestion des données:

La plateforme utilise une architecture de stockage hybride optimisée pour la performance et la scalabilité :

• **Stockage temporaire** : base de données Redis pour les données en temps réel (<24h), permettant des requêtes ultra-rapides pour les tableaux de bord.

• **Stockage historique** :

- Base de données MongoDB Atlas (cloud) pour l’archivage à long terme (>2 ans), avec indexation par capteur, date, et type de donnée.
- Politique de rétention configurable (ex : suppression automatique après 5 ans).

• **Sauvegarde et réplication** :

- Réplication géographique des données sur AWS S3 pour la redondance.
- Chiffrement AES-256 des données au repos et en transit.

4. Affichage des données en “temps réel”

• et d’effectuer des comparaisons entre différentes sources de données. L’affichage pourra inclure des graphiques avancés, des cartes interactives et des visualisations des tendances.

- Les utilisateurs (agriculteurs) pourront personnaliser les paramètres d’affichage en fonction de leurs besoins, par exemple en filtrant les données selon les critères souhaités (localisation, type de capteur, période, etc.).

L’interface utilisateur est conçue pour répondre aux besoins distincts des agriculteurs et des chercheurs :

- **Agriculteurs :**

Les agriculteurs disposent d’un tableau de bord simple et intuitif pour visualiser des informations essentielles:

- **Tableau de bord simplifié :**

- Visualisation de données clés (humidité du sol, température) via des graphiques linéaires et des jauges colorées.
- Cartes interactives montrant la localisation des capteurs et les alertes actives.
- Personnalisation des widgets (ex : ajout d’un graphique comparant l’humidité sur 7 jours).

- **Fonctionnalités tactiles :** Optimisé pour les tablettes et smartphones utilisés sur le terrain.

- **Chercheurs :**

Les chercheurs auront accès à un tableau de bord plus détaillé, permettant d’afficher des données complexes:

- **Interface analytique :**

- **Outils de superposition de données** (ex : comparer l’humidité du sol avec les précipitations historiques).
- **Visualisations avancées :** cartes heatmap, diagrammes de dispersion, séries temporelles annotées.
- Export de graphiques en haute résolution (PNG, SVG) pour les publications académiques.

5. Traitement des données

La plateforme propose des outils adaptés à chaque profil :

- **Agriculteurs : Calculs automatisés**

- Estimation des besoins en eau basée sur l’évapotranspiration et les prévisions météo.
- Alertes visuelles (icônes colorées) pour les seuils critiques (ex : réservoir d’eau à 20%).

- **Chercheurs : Analyse avancée**

- Modules statistiques intégrés (régression linéaire, analyse de variance) via une interface graphique.
- Intégration de modèles prédictifs personnalisables (ex : prédiction des rendements via Scikit-learn).



- Génération de rapports PDF automatisés avec résultats d'analyse et recommandations.

6. Alertes et notifications

Un système hiérarchisé permet une réactivité optimale.

Les utilisateurs seront notifiés en cas de seuils critiques dépassés. Les notifications pourront être envoyées par email ou notification push via la plate-forme, en fonction des préférences de chaque utilisateur.

Pour les chercheurs, des alertes seront également mises en place en cas de détection d'anomalies dans les données, facilitant ainsi une intervention rapide ou une analyse approfondie.

• Exemples de niveaux d'alerte :

- Urgent (rouge) : Détection d'une panne de capteur ou d'un seuil critique (ex : température < 0°C). Notification immédiate par SMS.
- Avertissement (orange) : Déviation modérée des normales saisonnières. Notification par email.
- Information (bleu) : Rappels périodiques (ex : rapport hebdomadaire de consommation d'eau).

• Personnalisation :

- Les agriculteurs peuvent définir leurs propres seuils via une interface simplifiée.
- Les chercheurs peuvent configurer des alertes basées sur des modèles complexes (ex : tendance à la baisse de la biodiversité).

7. Accessibilité et sécurité

• La plate-forme sera accessible via une interface web responsive, permettant aux utilisateurs d'y accéder depuis n'importe quel appareil (ordinateur, tablette, smartphone). Elle devra être compatible avec les principaux navigateurs web.

• Des niveaux d'accès différenciés seront mis en place :

• Les agriculteurs auront un accès limité à l'affichage des données et à la gestion de leurs propres dispositifs.

• Les chercheurs disposeront d'un accès plus étendu, incluant des outils d'analyse avancée et la possibilité de configurer des traitements de données.

• Les administrateurs auront des privilèges complets, notamment pour gérer les utilisateurs et paramétrer les capteurs et passerelles.

• Des mécanismes de sécurité seront mis en place pour garantir la confidentialité et l'intégrité des données. Cela inclut des protocoles de chiffrement pour les données transmises et stockées, ainsi que des systèmes de gestion des rôles pour contrôler les accès aux différentes parties de la plate-forme.

8. Scalabilité et performance

• La plate-forme sera conçue pour être scalable, afin de pouvoir accueillir une augmentation du nombre de capteurs et de passerelles, ainsi que de nouveaux utilisateurs dans le temps.



Elle devra être capable de supporter un volume croissant de données sans perte de performance.

- En ce qui concerne les temps de réponse, la plate-forme devra afficher les données en temps réel avec une latence minimale pour permettre une prise de décision rapide, en particulier pour les agriculteurs. Les fonctionnalités de traitement et d'analyse des données devront également être optimisées pour garantir une performance fluide même avec des volumes importants de données.

Aspects techniques

Contraintes techniques

Le développement de la plate-forme reposera sur une architecture modulaire et évolutive, capable de gérer les données provenant des capteurs IoT tout en offrant une interface accessible et sécurisée.

1. Architecture technique

La plate-forme sera basée sur une architecture client-serveur, avec une interface web pour l'affichage des données en temps réel et leur analyse.

- **Flux de données IoT :**

- Les capteurs transmettent les données aux passerelles via LoRaWAN ou Wi-Fi. Ces passerelles, déjà configurées dans Waziup, envoient les données vers le cloud Waziup via MQTT ou HTTP.
- Un service d'ingestion (Node.js) récupère périodiquement ces données brutes via les API Waziup, les transforme, et les stocke dans une base de données dédiée.
- Les données traitées sont exposées via une API Gateway (Python/Django) pour le frontend et les intégrations externes.

- **Couches principales :**

- **Frontend :** Application web responsive (React.js) hébergée sur un CDN pour réduire la latence.
- **Backend :**
 - Traitement en temps réel : Service Node.js pour l'agrégation des données et la gestion des alertes.
 - Traitement batch : Scripts Python (Apache Airflow) pour l'analyse historique et l'entraînement des modèles de ML.
- **Stockage :**
 - Temps réel : Redis pour la mise en cache des dernières mesures.
 - Historique : MongoDB Atlas (cluster shardé) pour gérer des téraoctets de données.

2. Technologies et outils

Les choix technologiques sont alignés sur les standards de Waziup et les besoins de scalabilité :

Composant	Technologies	Justification
Frontend	React.js + TypeScript, Chart.js, Leaflet	Interactivité élevée, compatibilité avec les API Waziup, et rendu cross-device.
Backend	Node.js (Express), Python (Django)	Intégration fluide avec les API REST de Waziup et gestion asynchrone des tâches.
Base de données	MongoDB Atlas (NoSQL), Redis	Flexibilité pour les schémas de données IoT et performances en temps réel.
Traitement de données	Pandas, NumPy, Scikit-learn, TensorFlow	Interopérabilité avec Python, utilisé par Waziup pour ses outils analytiques.
Cloud	AWS (EC2, S3, Lambda)	Hébergement élastique, compatible avec les services utilisés par Waziup.

3. Sécurité

La sécurité est structurée autour des normes adoptées par Waziup, avec des couches supplémentaires :

• Authentification :

- Utilisation de JSON Web Tokens (JWT) via l'API Auth de Waziup pour gérer les sessions utilisateurs.
- Rôles hiérarchisés (agriculteur, chercheur, admin) avec permissions granulaires (ex : un agriculteur ne peut accéder qu'à ses propres capteurs).

• Chiffrement :

- Données en transit : TLS 1.3 pour toutes les communications, y compris avec les API Waziup.
- Données au repos : Chiffrement AES-256 dans MongoDB Atlas et AWS S3.

• Audit :

- Journalisation centralisée des accès (ELK Stack) avec détection d'intrusions (ex : tentatives de brute force).



- Sauvegardes quotidiennes sur AWS S3, avec réplication cross-région.

4. Interfaces et intégrations

• API RESTful :

- Interne : API dédiée pour le frontend, documentée avec Swagger/OpenAPI.
- Externe : Intégration transparente avec :
 - Waziup : Synchronisation des capteurs et récupération des données via ses API.
 - Systèmes tiers : Export de données vers des outils d'analyse (ex : Power BI) ou des ERP agricoles.

• Protocoles supportés :

- MQTT (pour les passerelles IoT),
- HTTPS (pour les utilisateurs), et
- WebSocket (mises à jour en temps réel).

5. Scalabilité et performance

Afin de garantir une scalabilité optimale, la plate-forme sera hébergée sur des services cloud permettant d'adapter dynamiquement les ressources en fonction des besoins croissants.

• Scalabilité horizontale :

- Backend : Déploiement de conteneurs Docker sur AWS ECS, avec auto-scaling basé sur la charge (CPU > 70%).
- Base de données : Sharding automatique dans MongoDB Atlas pour répartir la charge.

• Optimisations :

- Cache : Redis pour réduire les appels redondants aux API Waziup.
- Compression : GZIP pour les données historiques (> 1 Mo).
- Latence : Objectif de < 500 ms pour les requêtes temps réel (testé avec JMeter).

6. Outils de gestion de projet

• Versioning :

- Git (GitHub) avec une stratégie GitFlow.
- Intégration des documentations techniques de Waziup dans le wiki du projet.

• Collaboration :

- Jira : Suivi des sprints, des bugs, et des dépendances avec les API Waziup.
- Slack : Communication en temps réel avec des canaux dédiés.

• CI/CD :

- Pipeline GitHub Actions / Jenkins pour les déploiements automatisés sur AWS, avec tests unitaires (Jest) et d'intégration (Postman).

7. Contraintes Liées à Waziup

- **Limitations techniques :**

- vTaux de requêtes API limité à 100 appels/minute. Contourné via une logique de cache et de requêtes groupées.
- Formats de données imposés pour l'enregistrement des capteurs (ex : {"type": "humidity", "unit": "%"}).

- **Dépendances :** Nécessité d'un abonnement actif à Waziup pour accéder aux API critiques.

La plateforme Waziup et son API

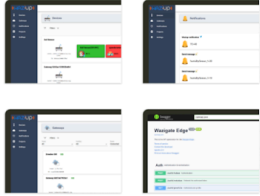
Description de waziup et son API

I. Présentation de Waziup

Waziup est une plateforme IoT open source spécialisée dans la gestion d'objets connectés pour l'agriculture et l'environnement. Son API RESTful (documentée via Swagger) permet de contrôler des capteurs, des passerelles, et des acteurs IoT, tout en offrant des outils pour le stockage, l'analyse et la notification des données.

Fonctionnalités principales :

- **Gestion centralisée des appareils IoT** (capteurs, actionneurs, passerelles).
- **Collecte et stockage de données** en temps réel avec historique.
- **Intégration flexible** avec des systèmes tiers via des API standardisées.
- **Sécurité** basée sur des tokens JWT et des rôles utilisateurs.



The WAZIUP Cloud platform allows you to manage your sensors, actuators and IoT data. WAZIUP Cloud platform offers everything that you need for your application:

- ✓ Remote connection of your sensors and actuators
- ✓ Send, receive, collect, store and analyze the data they generate
- ✓ Turn that data into actionable insights, in real time
- ✓ Using SMS or mobile application for notification
- ✓ Control your gateways from remote
- ✓ Extensive API for developing your own application

[Try WaziCloud](#) [Docs](#)

II. Structure de l'API Waziup (v2.0.0)

L'API est organisée en modules logiques, chacun couvrant un aspect spécifique de la gestion IoT. Elle peut être testée sur la page de la documentation officielle:

<https://api.waziup.io/docs>.

Voici une synthèse des endpoints clés :

1. Authentification (/auth)

- **POST /auth/token** : génère un token JWT pour accéder aux services protégés.

Exemple de requête : { "username": "admin", "password": "*****" }

Utilisation dans AgriLink : Ce token est indispensable pour interagir avec les capteurs et passerelles.

- **GET /auth/permissions/{ressource}** :



Liste les permissions de l'utilisateur pour une ressource (appareils, projets, etc.).

2. Gestion des Appareils (/devices)

- **POST /devices** : Crée un nouvel appareil IoT.

Exemple de payload : { "id": "sensor_123", "name": "Capteur Sol", "location": { "lat": 14.5, "lon": -17.5 } }

Utilisation dans AgriLink : Enregistrement initial des capteurs agricoles dans Waziup.

- **PUT /devices/{device_id}/location** : Met à jour la localisation géographique d'un appareil.

Exemple : Suivi des déplacements d'un capteur mobile.

3. Capteurs (/devices/{device_id}/sensors)

- **POST /devices/{device_id}/sensors** : Ajoute un capteur à un appareil.

Paramètres : sensor_kind (ex : "humidity"), unit (ex : "%").

Exemple : Ajout d'un capteur d'humidité à une station météo.

- **GET /devices/{device_id}/sensors/{sensor_id}/values** :

Récupère les données historiques d'un capteur.

Utilisation dans AgriLink : Alimentation des graphiques historiques pour les chercheurs.

- **POST /devices/{device_id}/sensors/{sensor_id}/value** : Envoie une valeur de capteur en temps réel.

Exemple : { "value": 45, "timestamp": "2025-03-01T12:00:00Z" }

4. Passerelles (/gateways)

- **POST /gateways** : Enregistre une nouvelle passerelle IoT (ex : une antenne LoRaWAN).

Exemple : Configuration d'une passerelle Multitech pour relayer les données des capteurs.

- **PUT /gateways/{gw_id}/heartbeat** : Met à jour le statut de santé de la passerelle.

Utilisation dans AgriLink : Surveillance des pannes réseau.

5. Notifications (/notifications)

- **POST /notifications** : Crée une règle de notification.

Exemple :

```
{
  "condition": { "sensor_id": "temp_01", "threshold": 40, "operator": ">" },
  "action": { "type": "email", "target": "alert@agrilink.org" }
}
```

Utilisation dans AgriLink : Alertes d'irrigation ou de gel pour les agriculteurs.



6. Projets (/projects)

- **POST /projects** : Crée un projet regroupant des appareils et passerelles.

Exemple : Projet "Ferme Pilote Dakar" liant 50 capteurs et 3 passerelles.

7. Ontologies (/ontologies)

- **GET /ontologies/sensor_kinds** : Liste les types de capteurs supportés (ex : "temperature", "humidity").

Utilisation dans AgriLink : Uniformisation des métadonnées des capteurs.

8. Données (/sensors_data)

- **GET /sensors_data** : Requête avancée pour filtrer les données par capteur, période, ou localisation.

Exemple : Extraction des données d'humidité sur juin 2025 pour une étude climatique.

9. Utilisateurs (/users)

- **POST /users** : Crée un compte utilisateur avec des rôles spécifiques (agriculteur, chercheur, admin).

III. Intégration avec AgriLink

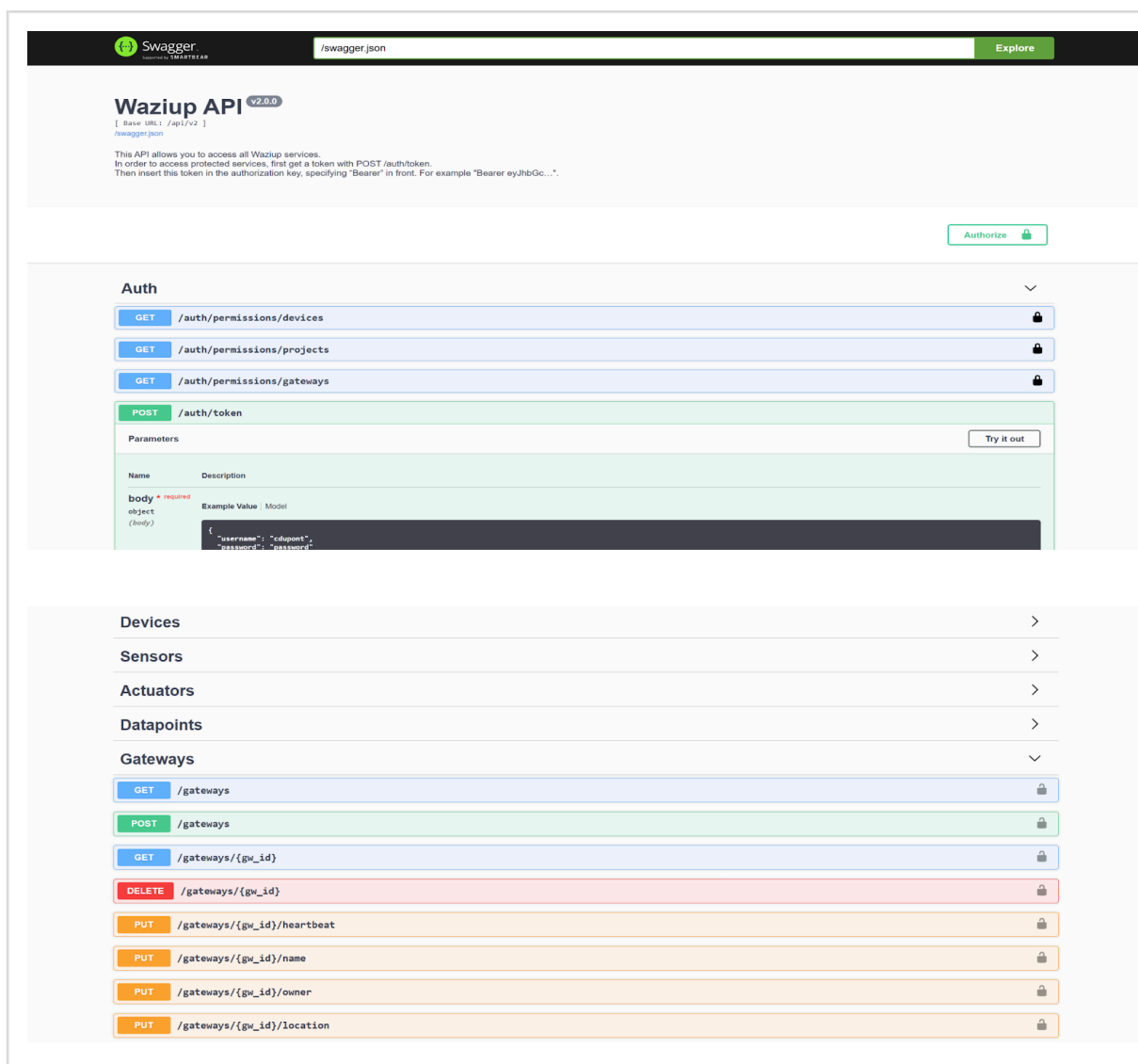
• Workflow typique :

1. Un capteur envoie des données à une passerelle Waziup via LoRaWAN.
2. La passerelle transmet les données au cloud Waziup via MQTT.
3. AgriLink récupère ces données via l'API /sensors_data pour affichage et analyse.
4. Les seuils d'alerte sont configurés via /notifications.

• Avantages clés :

- Interopérabilité : support natif des protocoles IoT (LoRaWAN, MQTT, HTTP).
- Scalabilité : gestion de milliers de capteurs sans surcharge.
- Open Source : personnalisation possible des workflows et extensions.





The image shows the Swagger UI for the Waziup API v2.0.0. The interface includes a top bar with the Swagger logo, the API name 'Waziup API v2.0.0', and a search bar containing '/swagger.json'. Below the top bar, there is a description of the API and an 'Authorize' button. The main content area is divided into sections for 'Auth' and 'Gateways'. The 'Auth' section lists endpoints for permissions and token generation. The 'Gateways' section lists endpoints for managing gateways, including GET, POST, DELETE, and PUT methods. A 'Try it out' button is visible next to the 'POST /auth/token' endpoint.

Waziup API v2.0.0
[Base URL: /api/v2]
/swagger.json

This API allows you to access all Waziup services.
In order to access protected services, first get a token with POST /auth/token.
Then insert this token in the authorization key, specifying "Bearer" in front. For example "Bearer eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXZWQ6...".

Auth

- GET /auth/permissions/devices
- GET /auth/permissions/projects
- GET /auth/permissions/gateways
- POST /auth/token

Parameters

Name	Description	Example Value	Model
body * required			
object (body)		{ "username": "edupont", "password": "password" }	

Devices

Sensors

Actuators

Datapoints

Gateways

- GET /gateways
- POST /gateways
- GET /gateways/{gw_id}
- DELETE /gateways/{gw_id}
- PUT /gateways/{gw_id}/heartbeat
- PUT /gateways/{gw_id}/name
- PUT /gateways/{gw_id}/owner
- PUT /gateways/{gw_id}/location

Ressources

Ressources

1. Ressources humaines

- **Chef de projet** : En tant que chef de projet, mes responsabilités incluent la coordination du projet, la gestion des ressources humaines, la planification et le suivi des délais, ainsi que la communication avec les parties prenantes. Je superviserai également l'intégration des différentes parties du projet.
- **Équipe de développement** : Le groupe de maîtres ingénieurs en informatique est constitué de 8 sous-groupes, chacun chargé de différentes parties du développement de la plate-forme.
- **Parties prenantes externes** : Les agriculteurs et chercheurs seront impliqués pour fournir des retours d'expérience et tester la plate-forme.



2. Ressources matérielles

- **Capteurs IoT** : Des capteurs pour mesurer la température, l'humidité, et les niveaux d'eau seront utilisés pour collecter les données en temps réel.
- **Passerelles IoT** : Les passerelles recevront les données des capteurs via des protocoles de communication (MQTT, HTTP) et les enverront à la plate-forme centrale.
- **Serveurs** : La plate-forme sera hébergée sur des serveurs cloud (par exemple, AWS) pour garantir l'évolutivité et la sécurité.
- **Équipements informatiques** : Les ordinateurs de bureau et portables utilisés par les équipes de développement pour coder et tester la plate-forme.

3. Ressources logicielles

- **Systèmes d'exploitation** : Les systèmes utilisés seront Linux pour les serveurs, et Windows ou macOS pour les postes de travail des développeurs.
- **Environnement de développement** : Visual Studio Code, PyCharm et IntelliJ IDEA seront utilisés pour le développement du frontend et du backend.
- **Langages de programmation** : Le développement du backend sera effectué en Node.js ou Python (Flask/Django), tandis que le frontend utilisera React ou Vue.js.
- **Frameworks et bibliothèques** :
 - Frontend : React.js, Bootstrap ou Material-UI
 - Backend : Flask, Node.js
 - Base de données : MongoDB ou PostgreSQL
 - Traitement des données : TensorFlow ou Scikit-learn pour les modèles prédictifs
 - Outils de gestion de projet : Nous utiliserons Jira pour le suivi des tâches, GitHub pour le versioning et le stockage du code source.

4. Ressources financières

Le budget du projet couvrira :

- L'achat des capteurs IoT et des passerelles nécessaires.
- L'hébergement sur des serveurs cloud pour la plate-forme.
- Les licences logicielles nécessaires, ainsi que les outils de développement.
- Le coût de la main-d'œuvre (étudiants et supervision).

Délais

Délais

Phase 1 : Spécifications et préparation (2 à 3 semaines)

Objectifs :

- Les étudiants commenceront par rédiger des spécifications détaillées pour la plate-forme (frontend, backend, et autres fonctionnalités).
- Définir les intégrations nécessaires entre les passerelles IoT existantes et la plate-forme.
- Déterminer les fonctionnalités à implémenter, comme l'affichage des données en temps réel, les outils d'analyse et de prise de décision pour les chercheurs, ainsi que les alertes et la gestion des ressources.

Livrables attendus :

- Document de spécifications pour la plate-forme, comprenant :
- Fonctionnalités principales de la plate-forme (affichage temps réel, analyse des données, gestion des alertes).
- Architecture technique de la plate-forme (choix des technologies, bases de données, APIs).
- Plan d'intégration des capteurs et passerelles existants avec la plate-forme.
- Schémas et diagrammes du flux de données entre les capteurs, passerelles et la plate-forme.
- Plan de tests et validation des fonctionnalités.
- Plan de gestion du projet incluant un calendrier des tâches et des jalons pour chaque groupe.

Répartition des tâches entre les groupes :

- Tâche 1 : Spécifications des interfaces utilisateur (UI/UX) pour les agriculteurs et les chercheurs.
- Tâche 2 : Spécifications du backend et intégration avec la base de données.
- Tâche 3 : Spécifications du système d'analyse des données et des prédictions pour les chercheurs.
- Tâche 4 : Spécifications des services de communication entre la plate-forme et les passerelles IoT.
- Tâche 5 : Spécifications des alertes et des notifications basées sur les données (ex : alertes pour l'irrigation).
- Tâche 6 : Spécifications de la gestion des utilisateurs et de la sécurité des données (authentification, cryptage).
- Tâche 7 : Spécifications des tests de performance (capacité à gérer plusieurs passerelles et de nombreux capteurs).
- Tâche 8 : Spécifications du processus de validation et des retours des utilisateurs finaux.

Durée : 2 à 3 semaines



Phase 2 : Développement de la plate-forme (8 à 10 semaines)

Objectifs :

- Développer la plate-forme digitale qui se connecte avec les passerelles IoT déjà installées.
- Chaque groupe commencera le développement des fonctionnalités qui lui sont attribuées, en respectant les spécifications validées lors de la phase 1.
- Les tests unitaires et d'intégration seront effectués au fur et à mesure du développement pour assurer la qualité du code et l'intégration des modules.

Livrables attendus :

- Modules fonctionnels développés pour chaque groupe (frontend, backend, analyse de données, alertes, etc.).
- Première version de la plate-forme intégrée avec les capteurs et passerelles IoT.

Planification des Sprints :

Sprint	Objectifs	Dates estimées	Livrables
Sprint 1	Développement du backend et intégration des passerelles IoT- Mise en place du backend pour stocker et gérer les données	Semaine 1 à 3	Backend fonctionnel, premiers tests d'intégration avec les passerelles IoT
Sprint 2	Développement des interfaces utilisateur (UI) pour les agriculteurs- Développement des API pour la gestion des données	Semaine 4 à 6	Interface utilisateur de base pour agriculteurs et chercheurs
Sprint 3	Développement des services d'analyse des données (prédictions, graphiques)- Intégration des alertes et notifications	Semaine 6 à 8	Services d'analyse en temps réel, alertes fonctionnelles
Sprint 4	Tests d'intégration et validation des fonctionnalités- Développement de la sécurité des données et gestion des utilisateurs	Semaine 8 à 10	Rapport de tests, corrections, sécurité et gestion des utilisateurs implémentées

Durée : 3 à 4 semaines



Phase 3 : Mise en production et documentation (3 à 4 semaines)

Objectifs :

- Préparer la plate-forme pour la mise en production.
- Fournir la documentation complète pour l'utilisation et la gestion du système.
- Organiser une session de formation pour les utilisateurs finaux (agriculteurs et chercheurs).

Livrables attendus :

- Version finale de la plate-forme prête à être déployée.
- Documentation complète : mode d'emploi pour les utilisateurs finaux et documentation technique.
- Rapport final sur le projet, avec une évaluation des performances et des retours des utilisateurs.

Durée : 3 à 4 semaines

Répartition du travail Scrum pour chaque groupe d'étudiants :

Chaque groupe se concentre sur des aspects spécifiques du projet, en lien avec l'intégration des données IoT dans la plate-forme.

1. Tâche 1 : Frontend pour les agriculteurs (affichage en temps réel des données, graphiques, gestion des alertes).
2. Tâche 2 : Backend et gestion des bases de données, intégration des capteurs et passerelles.
3. Tâche 3 : Analyse des données, prédictions et recommandations pour les chercheurs.
4. Tâche 4 : Intégration des passerelles IoT avec la plate-forme (communication et gestion des données).
5. Tâche 5 : Alertes et notifications pour les utilisateurs (gestion de l'irrigation, ressources, etc.).
6. Tâche 6 : Sécurité des données (authentification, cryptage, gestion des utilisateurs).
7. Tâche 7 : Tests de performance (capacité à gérer plusieurs capteurs et passerelles, latence).
8. Tâche 8 : Tests utilisateurs, validation des retours, et ajustements.

Suivi Scrum (Weekly Stand-ups, Sprint Reviews et Rétrospectives)

- Weekly Stand-up : Chaque groupe se réunit hebdomadairement pour partager son avancement, ses défis et ses priorités.
- Sprint Review : À la fin de chaque Sprint, les groupes présentent leur travail et reçoivent des retours pour affiner les fonctionnalités.
- Sprint Rétrospective : À la fin de chaque Sprint, une réunion est organisée pour discuter de ce qui a bien fonctionné et des améliorations possibles.



Budget (Proposition Alban)

Budget

Notre budget provisoire ci-dessous prend en compte les spécificités du projet, en privilégiant les outils open source et en excluant les coûts matériels (capteurs, passerelles).

Poste	Description	Description	Remarques
1. Hébergement Cloud	Services AWS pour héberger la plateforme (EC2, S3, MongoDB Atlas).	12 000	Coût estimé pour 6 mois, incluant stockage, calcul et base de données.
2. Outils de Développement	Environnements et logiciels nécessaires au codage et à la collaboration.	0	Utilisation de VSCode (gratuit) et GitHub (gratuit via GitHub Education).
3. Gestion de Projet	Outils pour le suivi des tâches et la coordination des équipes.	3 000	Licence Jira Cloud (Équipe ≤ 20 utilisateurs) pour 6 mois (500 MAD/mois).
4. Formation Utilisateurs	Sessions de formation pour agriculteurs et chercheurs.	10 000	5 ateliers (2 000 MAD/session) incluant supports pédagogiques et location de salles.
5. Rémunération Étudiants	Indemnités symboliques pour les 38 étudiants (8 équipes).	11 400	300 MAD/étudiant pour l'ensemble du projet (38×300).
6. Sécurité et Licences	Certificats SSL, outils de chiffrement.	2 500	Certificat Let's Encrypt (gratuit) + audit de sécurité ponctuel.
7. Contingences	Marge pour imprévus (10% du total hors contingences).	3 890	Calculé sur la somme des postes 1 à 6 ($38\,900 \text{ MAD} \times 10\%$).
Total		42 790 MAD	

Hypothèses et justificatifs

1. Hébergement Cloud :

- Coût basé sur un usage modéré (2 instances EC2, 500 Go de stockage S3, cluster MongoDB M30).
- Réduction possible via les programmes AWS Activate ou Azure for Students.

2. Gestion de Projet : Jira Cloud (Équipe) coûte ~500 MAD/mois pour environ 20 utilisateurs.

3. Économies Réalisées :

- GitHub : Utilisation du plan Éducation (gratuit) pour les dépôts privés et CI/CD.
- Waziup : Économie de ~20 000 MAD grâce à l'utilisation gratuite de ses API et outils IoT.

Répartition par phase

Phase	Coût (MAD)
Développement (6 mois)	28 000
Formation et déploiement	10 000
Contingences	3 890

Note : Ce budget est conçu pour être réaliste mais ne l'est pas pour l'instant. Les coûts pourront être ajustés en fonction des ressources disponibles

Budget (Proposition Groupe 8)

Budget

1. Coûts des ressources humaines

Ressource	Nombre	Coût unitaire	Durée (mois)	Coût total
Chef de projet	1	7 000 €/mois	4	28 000 €
Développeurs senior	4	5 500 €/mois	4	88 000 €
Développeurs junior	16	3 800 €/mois	4	243 200 €
Expert IoT	1	6 500 €/mois	2	13 000 €
Expert UX/UI	1	5 200 €/mois	2	10 400 €
Sous-total ressources humaines				382 600 €

2. Coûts matériels



Ressource	Nombre	Coût unitaire	Coût total
Serveurs pour environnement de développement	2	3 500 €	7 000 €
Équipements de test (ordinateurs portables)	8	1 200 €	9 600 €
Capteurs IoT supplémentaires pour tests	50	85 €	4 250 €
Passerelles IoT supplémentaires	5	350 €	1 750 €
Sous-total matériel			22 600 €

3. Coûts logiciels et services cloud

Ressource	Nombre	Coût unitaire	Durée	Coût total
Licences logicielles de développement	25	180 €/mois	4 mois	18 000 €
Services cloud AWS (développement)	-	1 500 €/mois	4 mois	6 000 €
Services cloud AWS (production - première année)	-	2 800 €/mois	12 mois	33 600 €
Base de données MongoDB Atlas (cluster production)	1	850 €/mois	12 mois	10 200 €
Outils de monitoring et d'analyse	1	350 €/mois	12 mois	4 200 €
Sous-total logiciels et services cloud				72 000 €

4. Coûts de formation et documentation

Ressource	Nombre	Coût unitaire	Coût total
Formation des utilisateurs (agriculteurs)	5 sessions	1 200 €	6 000 €
Formation des utilisateurs (chercheurs)	3 sessions	1 800 €	5 400 €
Documentation technique	1	8 500 €	8 500 €
Documentation utilisateur	1	6 800 €	6 800 €
Vidéos tutoriels	10	950 €	9 500 €
Sous-total formation et documentation			36 200 €

5. Coûts de maintenance et support (première année)

Ressource	Durée	Coût mensuel	Coût total
Support technique niveau 1	12 mois	3 200 €	38 400 €



Maintenance évolutive	12 mois	4 500 €	54 000 €
Mises à jour sécurité	12 mois	1 800 €	21 600 €
Sous-total maintenance et support			114 000 €

6. Coûts divers et imprévus

Ressource	Coût total
Déplacements et réunions	8 500 €
Imprévus (10% du budget total)	63 590 €
Sous-total divers et imprévus	72 090 €

7. Récapitulatif du budget total

Catégorie	Coût
Ressources humaines	382 600 €
Matériel	22 600 €
Logiciels et services cloud	72 000 €
Formation et documentation	36 200 €
Maintenance et support (première année)	114 000 €
Divers et imprévus	72 090 €
TOTAL	699 490 €

8. Modalités de paiement

- ☐ 30% à la signature du contrat
- ☐ 20% à la fin de la phase de spécification (Phase 1)
- ☐ 25% à la fin du développement (Phase 2)
- ☐ 15% à la mise en production (Phase 3)
- ☐ 10% après validation finale et acceptation du livrable

Délais détaillés

Phase 1 : Spécifications et préparation (3 semaines)

Semaine	Activités	Livrables
---------	-----------	-----------



S1	Analyse des besoins, étude technique initiale	Document d'analyse préliminaire
S2	Conception de l'architecture, définition des interfaces	Architecture technique détaillée
S3	Finalisation des spécifications, validation avec les parties prenantes	Document de spécifications validé

Jalons de la Phase 1 :

- Fin S1 : Validation de l'analyse des besoins
- Fin S3 : Présentation et validation des spécifications techniques

Phase 2 : Développement de la plate-forme (10 semaines)

Sprint 1 (Semaines 4-6):

- Backend : Développement de l'API RESTful
- Base de données : Mise en place du schéma MongoDB
- Intégration IoT : Configuration des passerelles

Sprint 2 (Semaines 7-9):

- Interface utilisateur : Développement du dashboard agriculteurs
- Intégration de données : Stockage et affichage en temps réel
- Sécurité : Mise en place de l'authentification

Sprint 3 (Semaines 10-12):

- Analyse de données : Algorithmes prédictifs
- Interface chercheurs : Développement des outils d'analyse avancés
- Système d'alertes : Configuration des seuils et notifications

Sprint 4 (Semaines 13-14):

- Tests d'intégration : Validation complète du système
- Optimisation des performances : Tests de charge
- Préparation de la mise en production

Jalons de la Phase 2 :

- Fin S6 : Démonstration de l'intégration IoT
- Fin S9 : Démonstration du dashboard agriculteurs
- Fin S12 : Démonstration des outils d'analyse pour chercheurs
- Fin S14 : Validation des tests d'intégration

Phase 3 : Mise en production et documentation (4 semaines)

Semaine	Activités	Livrables
S15	Déploiement en environnement de production, tests finaux	Plateforme déployée
S16	Rédaction de la documentation technique et utilisateur	Documentation technique
S17	Préparation et réalisation des formations utilisateurs	Support de formation
S18	Finalisation, préparation au lancement officiel	Rapport final du projet

Jalons de la Phase 3 :

- Fin S15 : Mise en production validée



- Fin S17 : Formation des utilisateurs clés
- Fin S18 : Présentation finale et lancement officiel

Calendrier global du projet

- **Début du projet** : 01/04/2025
- **Fin de la Phase 1** : 21/04/2025
- **Fin de la Phase 2** : 30/06/2025
- **Fin de la Phase 3 / Livraison finale** : 28/07/2025
- **Phase de support post-livraison** : 29/07/2025 - 29/07/2026

Gestion des retards

- Un retard de moins de 2 semaines sera géré par une intensification des ressources sans impact financier.
- Un retard entre 2 et 4 semaines nécessitera une révision du planning avec les parties prenantes.
- Un retard de plus de 4 semaines déclenchera la clause de pénalité (voir section Clauses contractuelles).

Clauses contractuelles

1. Confidentialité

Toutes les informations relatives au projet, y compris les spécifications techniques, les données des utilisateurs et les algorithmes développés, sont considérées comme confidentielles. L'équipe de développement s'engage à ne pas divulguer ces informations à des tiers sans autorisation écrite préalable.

2. Propriété intellectuelle

- Tous les droits de propriété intellectuelle sur les développements spécifiques réalisés dans le cadre du projet sont transférés au client.
- Les composants standards et bibliothèques open source utilisés restent sous leurs licences respectives.
- Le code source développé sera livré intégralement au client à la fin du projet.

3. Garantie et maintenance

Une garantie de bon fonctionnement de 12 mois est incluse à compter de la date de mise en production. Cette garantie couvre :

- La correction des bugs et anomalies
- Les mises à jour de sécurité
- Le support technique de niveau 1 et 2

4. Conditions de recette

La recette finale sera validée selon les critères suivants :

- Conformité aux spécifications fonctionnelles et techniques
- Performance du système (temps de réponse < 3 secondes pour 95% des requêtes)
- Disponibilité du système (99,5% minimum)
- Sécurité des données (validation par audit externe)



5. Pénalités de retard

En cas de retard imputable au prestataire :

- De 1 à 2 semaines : 1% du montant total du contrat par semaine de retard
- De 3 à 4 semaines : 2% du montant total du contrat par semaine de retard
- Au-delà de 4 semaines : 3% du montant total du contrat par semaine de retard, plafonné à 15% du montant total

6. Résiliation

Le contrat peut être résilié dans les cas suivants :

- Accord mutuel des parties
- Non-respect grave des obligations contractuelles après mise en demeure restée sans effet pendant 30 jours
- Retard de livraison supérieur à 8 semaines
- Force majeure persistant plus de 60 jours

7. Modalités d'évolution

Toute demande d'évolution fonctionnelle non prévue dans le présent cahier des charges fera l'objet d'une évaluation distincte et d'un avenant au contrat comprenant :

- Le détail des fonctionnalités supplémentaires
- L'impact sur les délais du projet
- Le coût additionnel
- Les conditions de mise en œuvre

8. Assurances

Le prestataire s'engage à souscrire une assurance responsabilité civile professionnelle couvrant les risques liés à l'exécution du projet pour un montant minimum de 1 000 000 €.

Spécifications techniques complémentaires

1. Architecture détaillée du système

1.1 Architecture frontend

- **Type d'application:** Single Page Application (SPA)
- **Framework:** React 18.x avec TypeScript 5.x
- **Gestion d'état:** Redux Toolkit ou Context API
- **UI/UX:** Material-UI 5.x ou Tailwind CSS 3.x
- **Visualisation de données:** D3.js, Recharts, ou Plotly
- **Tests:** Jest et React Testing Library

1.2 Architecture backend

- **Framework principal :** Node.js avec Express ou Python avec Django
- **API :** RESTful avec documentation Swagger/OpenAPI
- **Authentification :** JWT avec rotation des tokens et refresh tokens
- **Validation :** Joi ou Yup pour Node.js, Django REST framework validators
- **Tests :** Mocha/Chai pour Node.js, pytest pour Python

1.3 Base de données

- **Primaire :** MongoDB 6.x (NoSQL) ou PostgreSQL 15.x (SQL)
- **Cache :** Redis 7.x pour mise en cache des données fréquemment consultées
- **Time-series:** InfluxDB 3.x pour le stockage des données de capteurs



- **Backup** : Stratégie de sauvegarde quotidienne avec rétention de 30 jours

1.4 Architecture IoT

- **Protocoles supportés** : MQTT 5.0, HTTP/HTTPS, CoAP
- **Passerelle** : Architecture basée sur Eclipse Mosquitto ou AWS IoT Core
- **Traitement des données** : Apache Kafka ou AWS Kinesis pour le streaming
- **Edge computing**: Capacités de traitement local sur les passerelles

2. Spécifications de sécurité avancées

2.1 Sécurité des données

- Chiffrement des données sensibles en transit (TLS 1.3) et au repos (AES-256)
- Isolation des données par client avec cloisonnement strict
- Anonymisation des données utilisées pour l'analyse et les rapports agrégés
- Politique de rétention des données conforme au RGPD

2.2 Contrôle d'accès

- Authentification multi-facteurs pour les comptes administrateurs
- Système de rôles et permissions granulaire (RBAC)
- Journalisation des accès et des modifications (audit trail)
- Verrouillage des comptes après tentatives d'accès infructueuses

2.3 Sécurité de l'infrastructure

- Protection DDoS avec limitation de débit
- Pare-feu applicatif (WAF) pour protection contre les attaques web
- Analyse de vulnérabilités régulière (mensuelle)
- Tests d'intrusion annuels par un tiers certifié

3. Exigences de performance

3.1 Capacité et scalabilité

- Support jusqu'à 10 000 capteurs simultanés
- Traitement de 1 000 mesures par seconde
- Capacité à gérer 500 utilisateurs simultanés
- Temps de réponse inférieur à 2 secondes pour 95% des requêtes
- Disponibilité de 99,9% (hors maintenance planifiée)

3.2 Optimisation

- Mise en cache des données fréquemment consultées
- Pagination des résultats pour les requêtes volumineuses
- Compression des données pour optimiser la bande passante
- Lazy loading pour les composants frontend

4. Interface utilisateur détaillée

4.1 Dashboard agriculteurs

- Vue synthétique des indicateurs clés (humidité, température, niveau d'eau)
- Représentation cartographique des parcelles et capteurs
- Système de notification visuel pour les alertes (code couleur)
- Affichage des tendances sur 24h, 7j, 30j

- Mode hors ligne pour les zones à connectivité limitée

4.2 Interface chercheurs

- Outils d'analyse statistique avancée (corrélations, régressions)
- Export des données brutes et agrégées (CSV, JSON, Excel)
- Personnalisation complète des graphiques et visualisations
- Comparaison entre différentes parcelles/périodes
- Outils de modélisation prédictive avec paramétrage des variables

5. Fonctionnalités d'intelligence artificielle

5.1 Modèles prédictifs

- Prédiction des besoins en irrigation basée sur l'historique et les prévisions météo
- Détection précoce des maladies des cultures
- Optimisation de l'utilisation des ressources en eau

5.2 Détection d'anomalies

- Identification des comportements atypiques des capteurs
- Détection des défaillances matérielles potentielles
- Alertes précoces sur les variations environnementales anormales

5.3 Recommandations intelligentes

- Suggestions d'irrigation optimisée
- Recommandations sur les périodes de plantation/récolte
- Conseils adaptés aux conditions spécifiques de chaque parcelle

Annexes

Annexe 1 : Spécifications techniques des capteurs IoT existants

Type de capteur	Modèle	Précision	Fréquence d'échantillonnage	Autonomie	Portée
Humidité du sol	SM100	±3%	5 min	2 ans	500m
Température	TM200	±0.5°C	10 min	3 ans	800m
Niveau d'eau	WL150	±2mm	15 min	18 mois	600m
Station météo	WS500	Variable	30 min	Alimenté	1km



Annexe 2 : Spécifications techniques des passerelles IoT

Modèle	Connectivité	Capteurs max	Stockage local	Batterie de secours
GW-1000	Ethernet+4G	100	32 GB	12h
GW-2000	Ethernet+WiFi+4G	250	64 GB	24h

Annexe 3: Modèle de données préliminaire

// Exemple de structure de données capteur

```
{
  "sensor_id": "S12345678",
  "sensor_type": "soil_moisture",
  "location": {
    "latitude": 45.5017,
    "longitude": -73.5673,
    "field_id": "F12345"
  },
  "readings": [
    {
      "timestamp": "2025-03-15T14:30:00Z",
      "value": 37.5,
      "unit": "percent",
      "battery_level": 92
    }
  ]
}
```

Annexe 4 : Matrice de responsabilité (RACI)

Activité	Chef de projet	Équipe développement	Agriculteurs	Chercheurs
Spécifications fonctionnelles	R/A	C	C	C
Développement frontend	I	R/A	C	C
Développement backend	I	R/A	I	C
Tests utilisateurs	A	C	R	R
Mise en production	A	R	I	I
Formation	A	R	C	C

Légende : R-Responsable, A-Accountable, C-Consulted, I-Informed

Annexe 5 : Exemples d'interfaces utilisateur (maquettes)

[Description des maquettes d'interface qui seraient incluses ici]

Annexe 6: Plan de formation des utilisateurs

Public	Durée	Format	Contenu
Agriculteurs	1 jour	Présentiel	Interface basique, alertes, tableau de bord
Chercheurs	2 jours	Présentiel	Analyses avancées, export de données, modèles prédictifs
Administrateurs	3 jours	Présentiel	Configuration système, gestion utilisateurs, maintenance

Conclusion

Ce plan est maintenant axé sur l'intégration des capteurs IoT déjà réalisés avec la plate-forme digitale, avec des spécifications détaillées dès la première phase pour guider les groupes dans leur développement. La méthodologie Scrum permet une gestion itérative et flexible pour chaque groupe, tout en assurant une bonne coordination entre les équipes.