



Présentation du Projet

FOURNITURE D'UN SYSTÈME DE
MONITORING POUR UNE FLOTTE
DE VÉHICULES

BUGBROTHER
Nicolas ELFERING

24/01/2025

1. Soutenance du projet

- 1.1 Présentation du Projet
- 1.2 Systèmes embarqués
- 1.3 Applicatif
- 1.4 Infrastructure
- 1.5 Gestion de Projet
- 1.6 Bilan

2. Communication

3. Démonstration

4. Présentation individuelle



Soutenance du projet

Problématique: Monitoring d'une flotte automobile

- **Systèmes propriétaires par constructeur automobile :**
 - Tesla : Système intégré
 - BMW : ConnectedDrive
 - Mercedes-Benz : Mercedes me connect
 - Renault : My Renault
- **Problème principal:** Adaptées pour les propriétaires individuelles.
- **Objectifs:** Un système de supervision multimarque pour toutes les voitures conçus après 2002.

Présentation globale de l'architecture de la solution

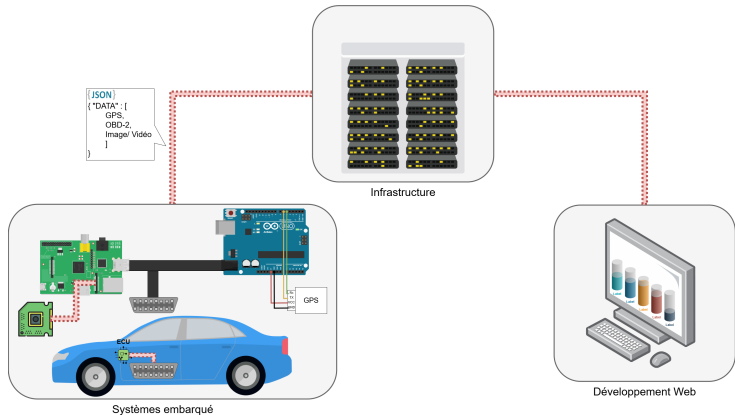


Figure: Schéma d'architecture globale de la solution

Principales fonctionnalités

- Gestion des flux vidéo et médias
- Suivi et monitoring des véhicules
- Statistiques et diagnostics
- Gestion des véhicules et des clients

Architecture des systèmes embarqués

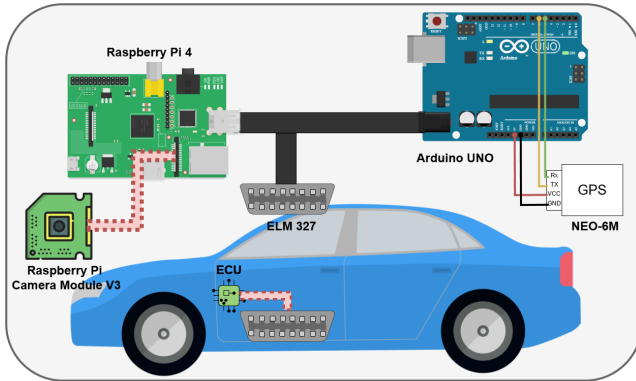


Figure: Architecture globale des systèmes embarqués.

Données collectées par les capteurs embarqués

- Raspberry pi 4 avec 3 threads :
 - Un thread pour le **GPS**.

Données collectées par les capteurs embarqués

- Raspberry pi 4 avec 3 threads :
 - Un thread pour le **GPS**.

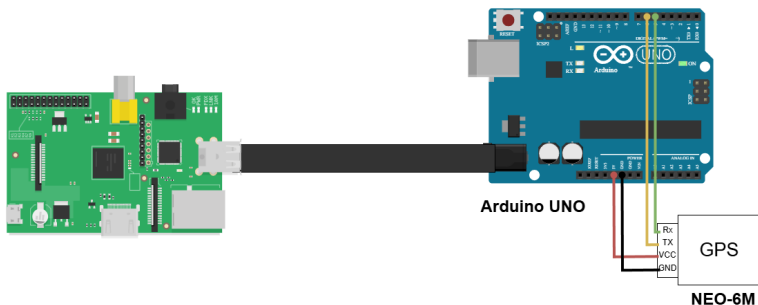


Figure: Récupération données GPS

Données collectées par les capteurs embarqués

- Raspberry pi 4 avec 3 threads :
 - Un thread pour le **GPS**.
 - Un thread pour l'**OBD-2**.

Données collectées par les capteurs embarqués

- Raspberry pi 4 avec 3 threads :
 - Un thread pour le **GPS**.
 - Un thread pour l'**OBD-2**.

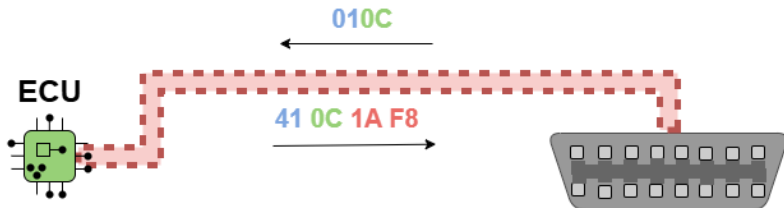


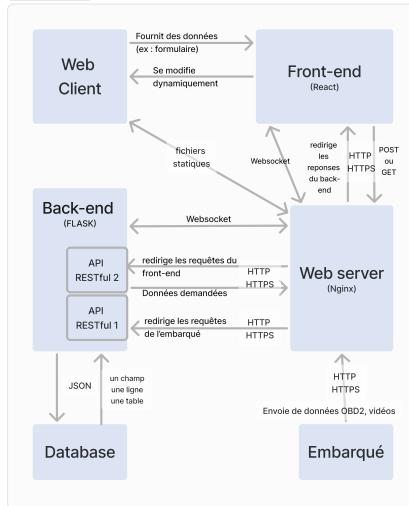
Figure: Récupération données OBD-2

Données collectées par les capteurs embarqués

- Raspberry pi 4 avec 3 threads :
 - Un thread pour le **GPS**.
 - Un thread pour l'**OBD-2**.
 - Un thread pour la **caméra embarquée**.
- Toutes les données dans la File du thread principal sont envoyées.

Vue globale de l'architecture applicatif

Schéma des flux



Pourquoi React ?

- Développement rapide.
- Adapté pour la création de PWA (Progressive Web App) :
 - Fonctionnement fluide en mode hors ligne.
 - Basculement automatique entre serveurs.

- **Pourquoi Flask ?**

- Minimaliste, flexible et adapté aux projets modulaires.
- Rapide à mettre en place avec une large communauté.

- **Organisation des API :**

- **Frontend API :** Requêtes utilisateur et données sécurisées.
 - Endpoints : `/api/frontend/...`
- **Embedded API :** Données GPS, OBD-2, et flux vidéo.
 - Endpoints : `/api/embedded/...`

- **Authentification :**

- Sécurisée via JWT (JSON Web Tokens).

Bases de données : Architecture et rôles

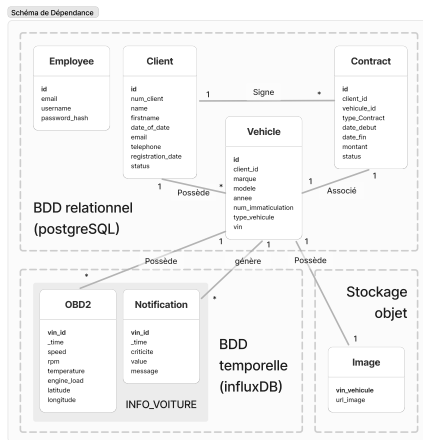


Figure: Schéma des interactions et rôles des bases de données.

Vue globale de l'infrastructure

Communications passant par le réseau internet

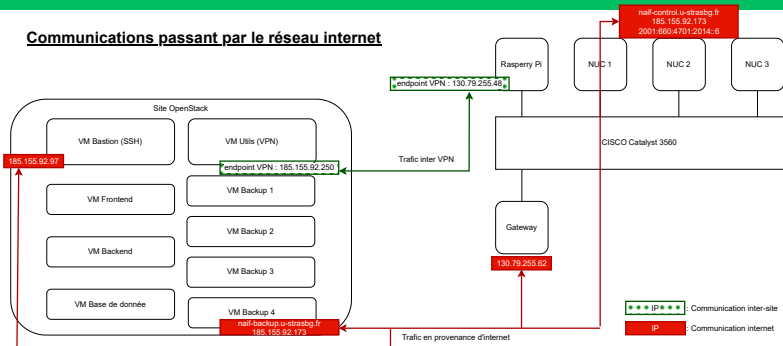


Figure: Architecture globale des deux sites.

- L'infrastructure repose sur deux sites complémentaires : **OpenStack (Esplanade)** et **C315 (Illkirch)**.
- Ces deux sites sont synchronisés en quasi-temps réel pour garantir la haute disponibilité et la résilience.

Site principal : C315 (Illkirch)

- **Rôle** : Site principal.
- **Caractéristiques techniques** :
 - Virtualisation via Proxmox.
 - Hébergement des VM: BDD, backend et frontend.
 - Gestion des IP flottantes.
- **Points forts** :
 - Stockage CEPH, Haut disponibilité.
 - monté en charge et monitoring. item Interconnexion VPN avec OpenStack pour synchronisation BDD.
 - prise en charge d'IPv6

Site secondaire : OpenStack (Esplanade)

- **Rôle** : Data-center secondaire.
- **Caractéristiques techniques** :
 - Virtualisation via OpenStack.
 - Hébergement des VM: BDD, backend et frontend pour le **développement et la production**.
 - Gestion des IP flottantes.
- **Points forts** :
 - Interconnexion VPN avec C315 pour synchronisation BDD.
 - monté en charge et monitoring.

Monitoring et montée en charge de l'infrastructure

- **Outils utilisés :**

- **Prometheus** : Collecte des métriques des serveurs et services.
- **Grafana** : Tableaux de bord interactifs pour visualiser les performances.
- **Nomad** : Orchestration des tâches et gestion dynamique des ressources.
- **Consul** : Équilibrage du trafic et découverte des services.

- **Fonctionnalités principales :**

- Détection des anomalies et alertes en temps réel.
- Suivi des performances entre OpenStack et C315.
- Répartition automatique des tâches en cas de surcharge.

- **Montée en charge :**

- Ajout dynamique de conteneur via Nomad.

- **Répartition des rôles :**

- Chef de projet : Coordination globale et suivi des jalons.
- Équipe infrastructure : Gestion des serveurs (OpenStack, C315).
- Équipe systèmes embarqués : Développement du Raspberry Pi et capteurs.
- Équipe développement web : Création du frontend, backend et bases de données.

- **Répartition des rôles :**

- Chef de projet : Coordination globale et suivi des jalons.
- Équipe infrastructure : Gestion des serveurs (OpenStack, C315).
- Équipe systèmes embarqués : Développement du Raspberry Pi et capteurs.
- Équipe développement web : Création du frontend, backend et bases de données.

- **Outils utilisés :**

- **YouTrack** : Gestion du temps, répartition des tâches et diagrammes de Gantt.
- **GitLab** : Sauvegarde du code et gestion des versions.
- **Discord** : Communication et réunions en visioconférence.

- **Phases principales du projet :**

- Phase 1 : Analyse des besoins et conception (1 mois).
- Phase 2 : Développement des systèmes embarqués, web, et infrastructure (2 mois).
- Phase 3 : Intégration et tests (1 mois).

- **Phases principales du projet :**

- Phase 1 : Analyse des besoins et conception (1 mois).
- Phase 2 : Développement des systèmes embarqués, web, et infrastructure (2 mois).
- Phase 3 : Intégration et tests (1 mois).

- **Livrables clés :**

- Maquette initiale.
- Rendu intermédiaire.
- Rendu final.

- **Phases principales du projet :**

- Phase 1 : Analyse des besoins et conception (1 mois).
- Phase 2 : Développement des systèmes embarqués, web, et infrastructure (2 mois).
- Phase 3 : Intégration et tests (1 mois).

- **Livrables clés :**

- Maquette initiale.
- Rendu intermédiaire.
- Rendu final.

- **Suivi du diagramme de Gantt :**

- Mise à jour hebdomadaire.

- **Phases principales du projet :**

- Phase 1 : Analyse des besoins et conception (1 mois).
- Phase 2 : Développement des systèmes embarqués, web, et infrastructure (2 mois).
- Phase 3 : Intégration et tests (1 mois).

- **Livrables clés :**

- Maquette initiale.
- Rendu intermédiaire.
- Rendu final.

- **Suivi du diagramme de Gantt :**

- Mise à jour hebdomadaire.

- **Défis rencontrés :**

- Gestion des délais.
- Une tâche n'était pas assez précise.

- **Points forts :**

- Infrastructure robuste avec redondance et monitoring efficace.
- Systèmes embarqués fonctionnels, intégrant GPS, OBD-2 et caméra.
- Application web complète avec frontend, backend, et bases de données intégrées.
- Objectifs principaux atteints :
 - Gestion des flux vidéo et médias.
 - Suivi et monitoring des véhicules.
 - Statistiques et diagnostics.
 - Gestion des véhicules et des clients.

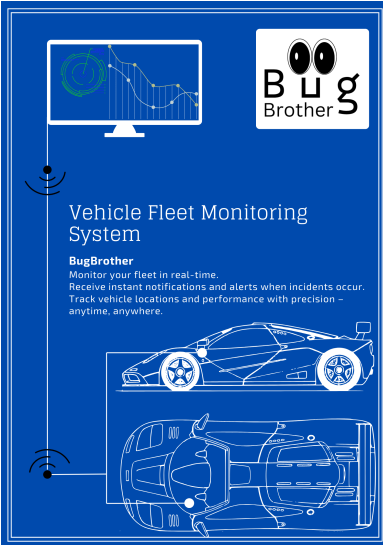
- **Limites :**

- PWA pas fini.



Communication

- Un compte Instagram est utilisé comme vitrine pour montrer nos capacités et le produit proposé.
- Nous pensons qu'un compte pour les réseaux sociaux ne serait pas suffisant pour atteindre efficacement notre cible.

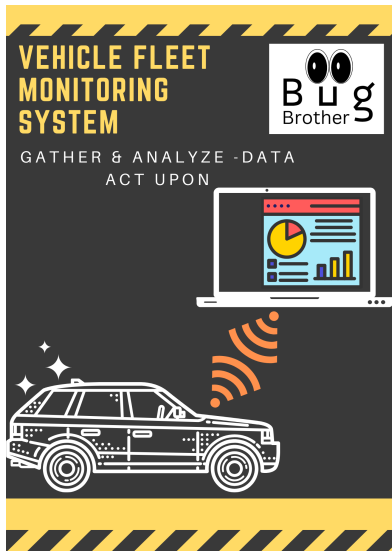


The advertisement is set against a dark blue background. At the top left, a computer monitor displays a line graph with a green circular arrow icon. To its right is the 'Bug Brother' logo, which features two large eyes above the text 'Bug Brother'. Below the monitor, a vertical line with two wireless signal icons connects to two car illustrations. The top car is a side profile of a sports car, and the bottom car is a top-down view of a similar vehicle. Text is positioned between the cars.

Vehicle Fleet Monitoring System

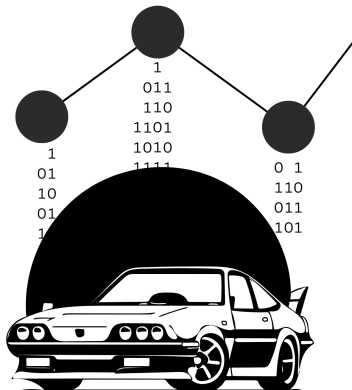
BugBrother
Monitor your fleet in real-time.
Receive instant notifications and alerts when incidents occur.
Track vehicle locations and performance with precision – anytime, anywhere.

Affiches pour événements



Gather & analyze -data
act upon

Bug
Brother



VEHICLE FLEET MONITORING SYSTEM

- Une vidéo publicitaire a été créée, conçue pour être diffusée à la télévision.
- Visionnez la vidéo ici : **Lien vers la vidéo publicitaire**



Démonstration



Présentation individuelle

Gestion de l'infrastructure système et réseau du site OpenStack

- Environnement de développement
- Environnement de secours
- Machines utilitaires

DevOps

- Automatisation de la configuration des machines
- Pipeline d'intégration continue
- Définition de jobs Nomad

Communications avec l'extérieur

- Interconnexion des sites
- Accessibilité depuis Internet

Gestion des bases de données du site OpenStack

- Déploiement des bases de données
- Synchronisation des bases de données

Sécurité sur le site OpenStack

- Définition des règles de pare-feu
- Création du bastion

Analyse et documentation

- Évaluation de performances
- Documentation et cartographie

Gestion du site C315

- Déploiement et configuration de Proxmox
- Configuration réseau
- Outillage

Gestion inter-sites

- Connectivité inter-sites
- Déploiement et jobs Nomad et Consul

Vie de l'infrastructure

- Rédaction de procédures
- Automatisation
- Réponse à incident

Conception et développement du matériel embarqué

- Collecte des données
 - GPS
 - OBD-2
- Envoi des données à l'API
- Communication entre l'Arduino et le Raspberry Pi

Infrastructure

- Développement et mise en production de jobs Nomad

Développement sur le Raspberry Pi pour le module caméra

- Envoi périodique de photos
- Enregistrement vidéo continu des 30 dernières secondes
- Envoi d'un flux vidéo en temps réel

Choix et mise en place de l'architecture pour le streaming

- Choix du protocole de streaming : Real Time Streaming Protocol (RTSP) et HTTP Live Streaming (HLS)
- Mécanisme de notification
- Installation d'un serveur de streaming : MediaMTX
- Utilisation du Vehicle Identification Number (VIN) dans les endpoints

Développement applicatif

- Création des pages d'authentification, tableau de bord, contrats
- Intégration de la carte interactive pour visualiser les véhicules
- Optimisation des interfaces utilisateur

Backend et communication

- Mise en place de l'API entre React et Flask
- Authentification sécurisée via JWT

Gestion des données et UX

- Intégration d'InfluxDB et PostgreSQL
- Améliorations UX : thème sombre, sider, pages d'erreur personnalisées

Développement applicatif

- Conception de la maquette Figma
- Création des pages ajout, véhicule
- Gestion des erreurs sur les formulaires
- Mise en place du responsive design

Backend et communication

- Mise en place de l'API entre React et Flask

Gestion des données

- Création et gestion de PostgreSQL
- Système de notifications, via InfluxDB

Gestion de projet

- Coordination en tant que chef de projet.
- Gestion du temps et suivi à l'aide de diagrammes de Gantt.

Communication et supports visuels

- Création de la vidéo publicitaire avec Blender.
- Gestion des éléments de communication pour les événements.

Contributions techniques

- Mise en place des WebSockets pour la communication en temps réel.
- Assistance au développement de l'OBD-2.
- Configuration du stockage MinIO (S3).
- Support à l'intégration des fonctionnalités vidéo.



Des questions ?