

watts UP

CAMILLE CÉLESTE COVAREL

DWWM · décembre 25

Watts Up ?

Watts Up est une plateforme de réservation de bornes de recharge pour les usagers de véhicules electriques.

Lineup



Jonathan Dias

*Dev FullStack / Scrum Master
orienté front-end*



Camille Céleste Covarel

*Dev FullStack / Product Owner /
Tech Lead
orienté back-end*

Pour quels besoins ?

En tant que visiteur

Découvrir le service

Voir les stations disponibles

En tant que client

Réserver une borne

Se géolocaliser

Avoir son espace personnel

Contacter le service client

En tant qu'administrateur

Gérer les stations / bornes

Gérer les clients

On travaille pour quoi ?

Créer un MVP à présenter à la
société fictive GeoCode

Stack technique

FrontEnd

Vite.js
Maplibre
TanStack
Query

BackEnd

Node.js
Express
Sequelize
+ cli

BDD

Postgresql
Postgis

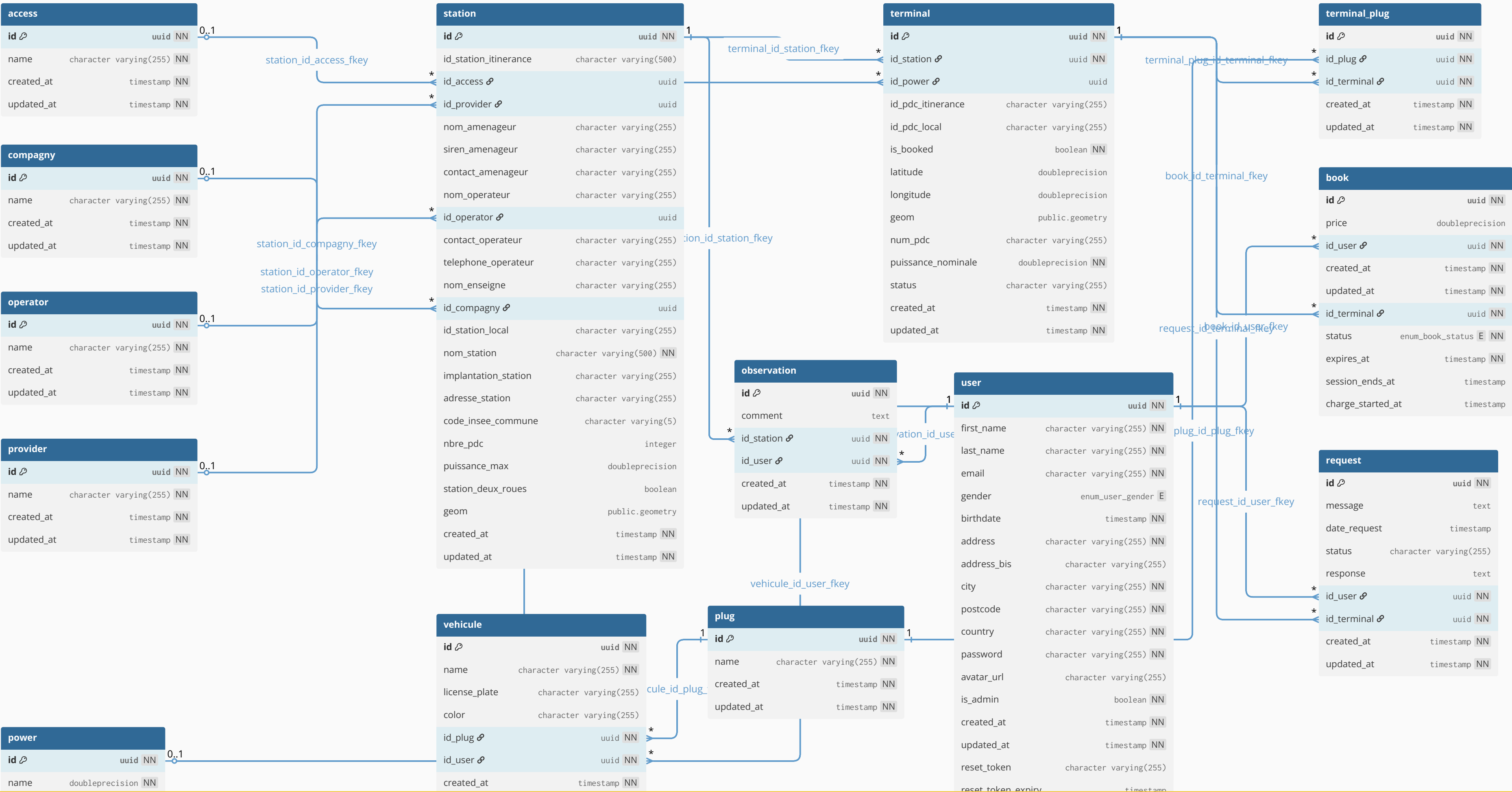
Déploiement

Docker
Github
GHCR
Github action
Nginx

Maquettage de l'app



Modélisation BDD



PostgreSQL + PostGIS

Après analyse de la data, veille
et discussion avec Tristan Laroye,
PostGIS a été une évidence.

Addon spécialisé dans les
coordonnées géospatiales, PostGIS
nous a permis de déployer des tuiles
vectorielles.

MongoDB + GeoJSON

Requêtes géospatiales natives mais
très différent niveau structure.

Elasticsearch Geo

Performant mais overkill et lourd

PostgreSQL + PostGIS

Migration nécessaire mais tuiles
vectorielles donc très bonne
performance à peu de charge.

Import CSV, Le défi

Mon gros défi sur ce projet. Importer plus de 170k lignes (35.5k stations, 136k bornes). J'ai réalisé cette tâche en suivant les préceptes des transactions ACID

atomicité, cohérence, isolation et durabilité

Traitement ligne par ligne

fs.createReadStream

~ ~ ~ ~ ~

Validation et clean de la data

dataTransformer.ts

~ ~ ~ ~ ~

Consolidation en mémoire

Stations avec bornes pré-validées dans une map
(les relation simples sont gérés par un findOrCreate)

~ ~ ~ ~ ~

Mise en cache

Pour éviter les transactions inutiles, la fonction sait où elle en est

~ ~ ~ ~ ~

5000 lignes or 2000 stations

Une fois atteint ce pallier, transaction ACID vers BDD

pages 18 à 19 du dossier de projet

Transaction & Rollback

Si un terminal ou un autre élément
échoue, la transaction est annulée.
l'erreur est loggée dans une fonction
custom.

Cela évite d'avoir des données
incomplètes et protège la BDD.

page 20
à 21 du
dossier de
projet

Monitoring WebSockets

L'administrateur possède une vision directe sur cet import. Grâce aux WebSockets, il voit en direct :

La progression de l'import

Les erreurs

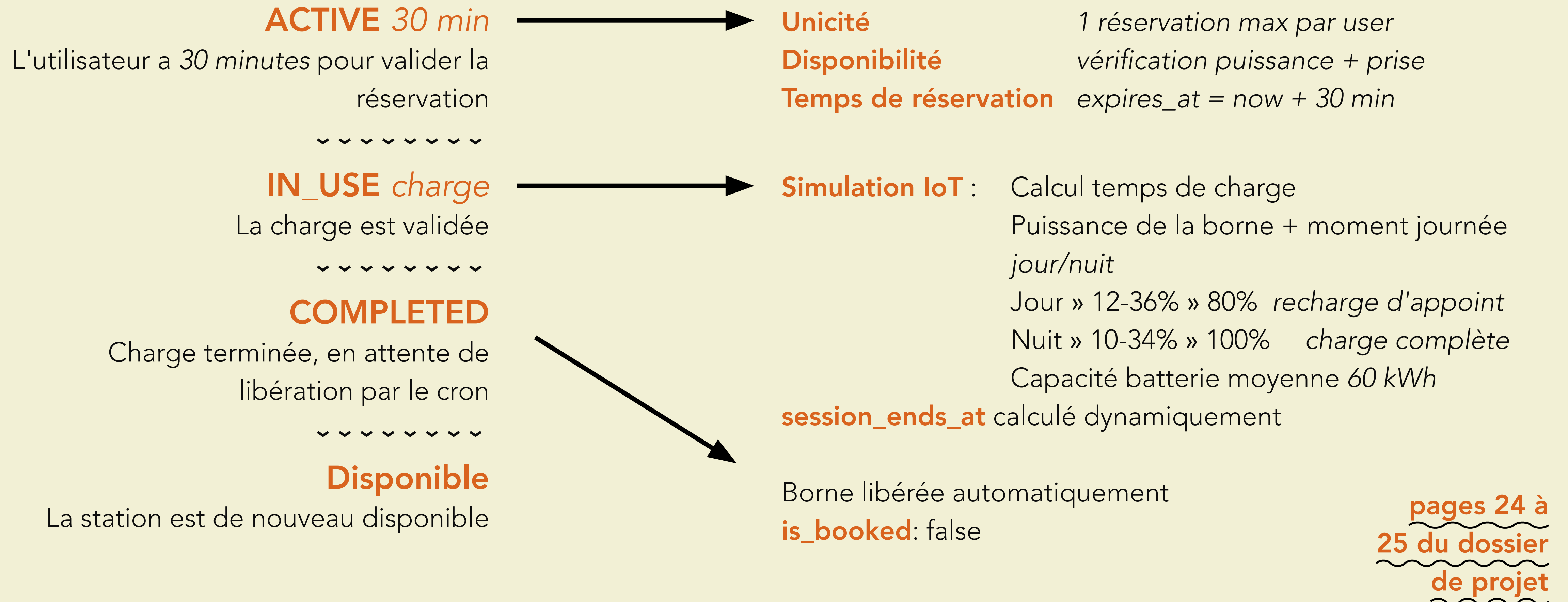
La durée

Le statut final

Une fonction de réhydratation est également présente pour qu'il puisse récupérer les informations à n'importe quel moment.

page 22
du dossier
de projet

Systeme de reservation



Architecture adaptative

watts UP

Jean Jaurès
16 allée Jean Jaurès 31000 Toulouse FRANCE

Implantation: Parking public
Horaires: 24/7
Accès PMR: Oui
Points de charge: 67

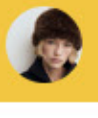
Choisir ma borne

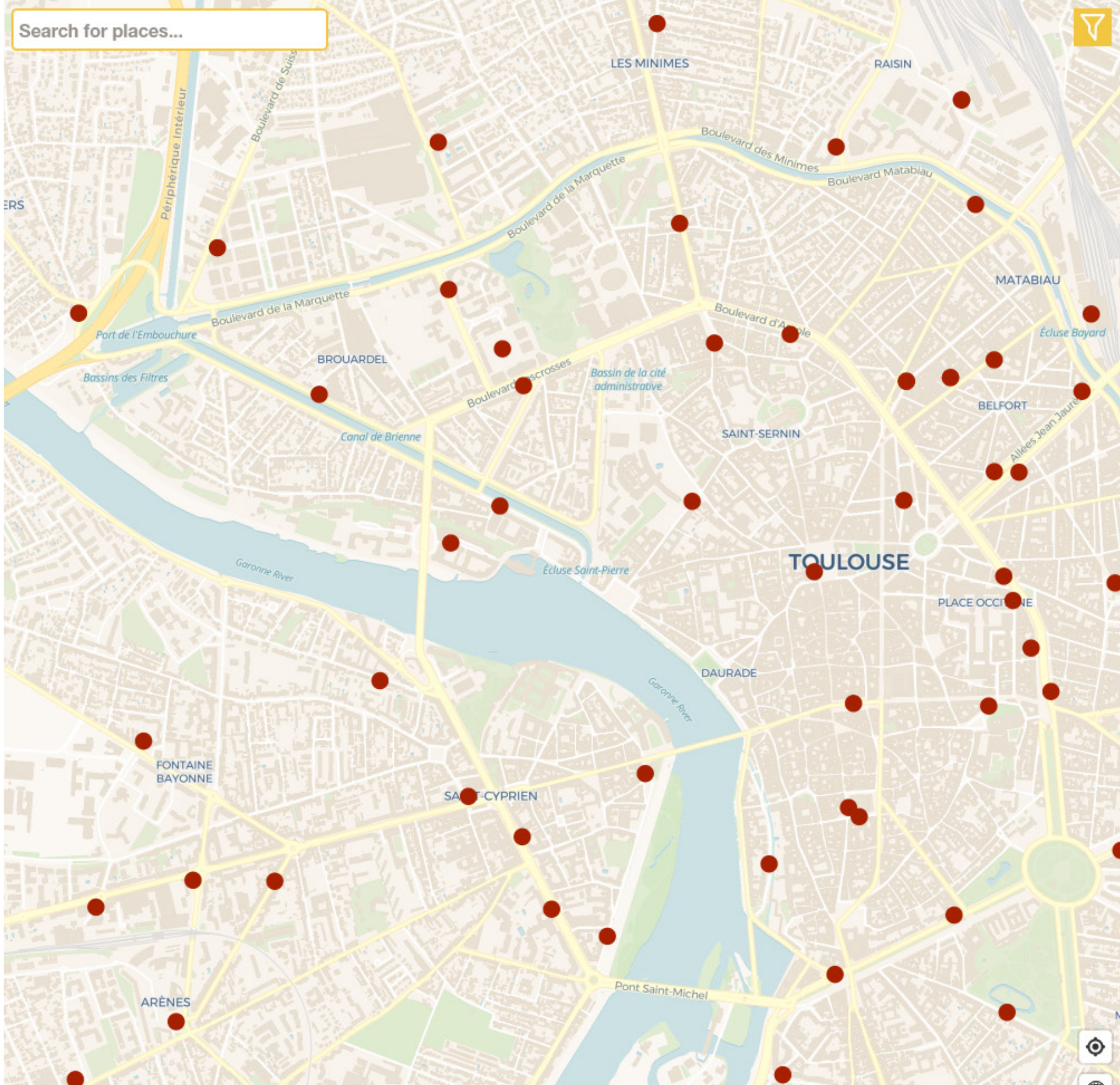
**7.4 kW**
60/60

3.7 kW
7/7



watts UP


Carte Mes réservations Assistance Informations Dashboard Admin 




Parc Victor Hugo
Place Victor Hugo 31500 Toulouse FRANCE

Implantation: Parking public
Horaires: 24/7
Accès PMR: Oui
Points de charge: 26

Choisir ma borne

**7.4 kW**
19/19

**3.7 kW**
1/1

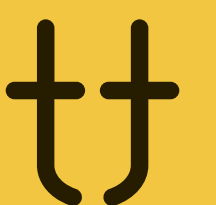
3.7 kW
6/6

Réserver

Retour à la carte

Supprimer la station

page 28
à 30 du
dossier de
projet



Cycle de vie réservation

Pour plus de clarté, les bornes sont agrégées côté client avec useMemo sous le format

22 kW - Type 2 - 1/8
puissance - type prise - dispo/total

Les timers sont en temps réel
Avec le status ACTIVE, on a un compte à rebours

Avec le status IN_USE, on a un chronomètre *setInterval* avec l'heure de fin de charge
Simulation IoT

page 32
du dossier
de projet

Feedback utilisateur/erreurs

L'application possède des Toasts et Modales branchés directement sur les mutations de TanStack Query

Les Toasts sont utilisés pour les erreurs et succès

Les Modales sont utilisés pour les actions sensibles avec validation explicite
réservation, annulation

Un autre type de feedback utilisateur concerne les alertes affichées en surimpression

Fichier trop volumineux (max 1 Mo)

pages 32 du
dossier de
projet

watts UP

DÉMO LIVE

watts-up.anamnesis.ovh

Jeux d'essai

<u>Sélection</u>	<u>Réservation</u>	<u>Charge</u>	<u>Double réservation</u>	<u>Mauvaises données</u>
stations madrid v v v	tentative réservation v v v	démarrage charge v v v	seconde réservation v v v	rejet de la ligne + log v v v
bornes affichées v v v	minuteur 30 minutes v v v	chrono actif v v v	blocage + message v v v	rejetée et loggée v v v
nominale	nominale	nominale	nominale	nominale

page 35 à 41 du dossier de projet

Veille & Sécurité

Maplibre vs Leaflet

Rendu GPU / Rendu DOM

Meilleure fluidité pour MapLibre

Sequelize vs Prisma vs SQL

Le SQL vanille n'était pas envisageable au vu de l'équipe et du projet. Prisma a un langage propre. Sequelize nous assurait un traitement sécurisé et rapide de l'application.

PostGis

MapLibre

Sequelize

Sequelize-Cli

UUID

JWT httpOnly

Rate Limiting

tuiles vectorielles

rendu GPU performant

anti-injection SQL

migration versionnée

anti-énumération

anti-XSS

anti-brute force

pages 46 à
47 du dossier
de projet

Déploiement & Bilan

GitHub Actions CI

Lance l'intégration quand **PR** ou **Push** sur
branche staging

~ ~ ~ ~ ~ ~ ~ ~

Build

Tests & images versionnées
(SHA commit)

~ ~ ~ ~ ~ ~ ~ ~

Déploiement CD

21 secrets sont nécessaire pour
déployer l'application **sur 2 CTs**

~ ~ ~ ~ ~ ~ ~ ~

Disponible

l'application est disponible

Technique

L'application est déployée automatiquement en production
et est fonctionnelle. La solution est performante.

Apprentissages

Une belle réussite sur le traitement de la data.

La découverte de PostGis et PostgreSQL avec maplibre.

Perspective d'évolution

Ajout de tests pour chaque feature.

Ajout des refresh token et du 2FA.

Mise en conformité RGPD.

etc.

page 42,43
et 50 du
dossier de
projet

watts UP

MERCI