opquast

Rendre le web meilleur

Webinaire : Améliorer vos services numériques avec Opquast et ses outils IA

Webinaire - 26 mars 2026

Les intervenants

Benjamin Hannache
Directeur Général Opquast

Julie Morana
Cheffe de projet - Opquast

Mickael Hoareau
Responsable technique - Opquast

Objectifs du webinaire

1 Introduction : l'IA pour améliorer la qualité des services numériques

1 Présenter le MCP d'Opquast (version Bêta)

2 Faire un retour d'XP sur l'utilisation du MCP

3 Présenter quelques éléments de roadmap !

Au programme

1. Introduction - vue d'ensemble	10 min
2. MCP Opquast : présentation	20 min
3. MCP Opquast : retour d'XP (📣)	15 min
4. Roadmap & Conclusion	15 min

L’IA dans les activités du Delivery

Cycle de delivery simplifié

Chef de projet / Scrum Master

Réunions -> CR
Analyses KPI
Planning
Doc d’organisation

Product Owner / Business Analyst

Ateliers -> CR
Analyses KPI
User Stories
Tests

UX/UI Designers

Ateliers -> CR
Evaluation
Wireframes
Maquettes UI
Illustrations
Chartes
Documentation

Développeurs

Ateliers
Coding
Tests
Débug
Déploiement
Documentation
Exploitation

Rédacteurs

Charte
Rédaction
Illustration
Edition / Relecture
Traduction

Taux d’utilisation de l’IA (observations qualitatives)

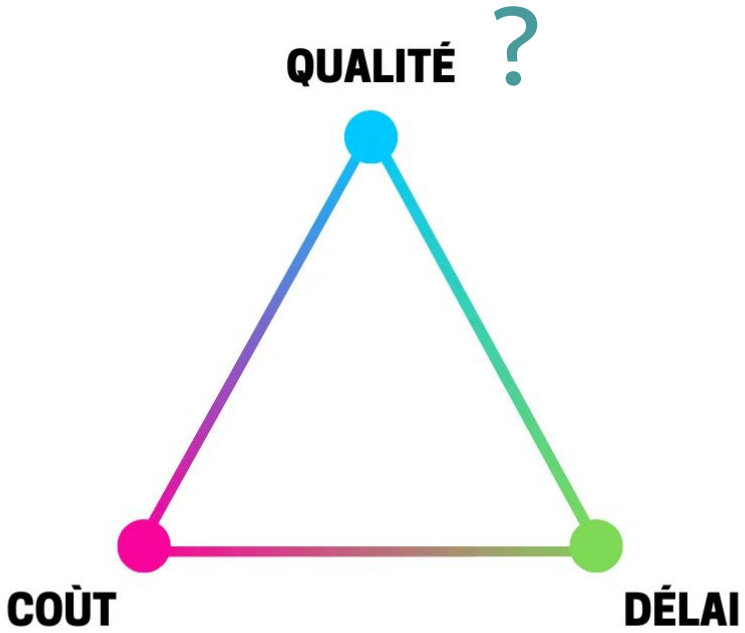
L’IA dans les activités du Delivery

Product Builder
(ou Full Stack Creator ou Vibe Builder…)

Chef de projet / Scrum Master	Product Owner / Business Analyst	UX/UI Designers	Développeurs	Rédacteurs
				
Réunions -> CR Analyses KPI Planning **Doc d’organisation**	Ateliers -> CR Analyses KPI **User Stories** Tests	Ateliers -> CR Evaluation Wireframes **Maquettes UI** **Illustrations** Chartes Documentation	Ateliers **Coding** **Tests** **Débug** **Déploiement** **Documentation** **Exploitation**	Charte **Rédaction** Illustration **Edition / Relecture** **Traduction**

 Taux d’utilisation de l’IA (observations qualitatives)

L’IA dans les activités du Delivery



Pour démarrer : demander, tout simplement

Formuler une demande en langage naturel, les IA (très) récentes partiront de là pour proposer des outils :

Je voudrais installer tout ce qui est à ta disposition pour améliorer le code que tu produis au niveau qualité, sécurité, impact environnemental performance, protection des données personnelles, accessibilité, UX.

Tu peux me suggérer des MCP, un fichier de configuration, la prise en compte de mon contexte, la prise en compte des règles opquast.

Une fois qu'on aura fait cela, on relancera la fabrication du site de mon amap.

Lui donner nos standards

Lui donner nos check-lists

Lui donner nos critères d'évaluation

Lui donner notre expertise.

Interlude : les règles Opquast

opquast

245 règles - Licence ouverte - commun numérique

checklists.opquast.com/fr/qualite-numerique

Référentiel Opquast pour les services numériques – 14 rubriques

Contenus 14	Images et médias 12	Présentation 17
Données personnelles 15	Internationalisation 8	Sécurité 21
E-Commerce 39	Liens 17	Serveur et performances 13
Formulaires 30	Navigation 7	Structure et code 15
Identification et contact 17	Newsletter 17	

Légende :
X Nombre de règles

Des fichiers d’instructions par thèmes

Vous pouvez (devez) dire à l'IA ce que vous attendez.

```
agent_docs/
  |- building_the_project.md
  |- running_tests.md
  |- code_conventions.md
  |- service_architecture.md
  |- database_schema.md
  |- service_communication_patterns.md
```

A savoir : plus on donne d’instructions, plus elles sont susceptibles d’être ignorées

Travailler en liste de tâches et en planification

Update Todos

- [*] Installer PHP et Composer via Homebrew
- [ ] Initialiser Git et creer .gitignore
- [ ] Creer composer.json et installer PHPCS + WordPress Coding
- [ ] Creer package.json et installer ESLint, Stylelint, Prettier
- [ ] Creer les fichiers de config linters (.eslintrc.json, .stylelintrc.j
- [ ] Creer .editorconfig
- [ ] Modifier wp-config.php (debug constants)
- [ ] Creer CLAUDE.md
- [ ] Extraire et creer la checklist Opquast (docs/opquast-checklis
- [ ] Configurer Claude Code (.claude/settings.json + .mcp.json)
- [ ] Verification finale de tout l'environnement

Étape 5 : Charger les CSS des shortcodes

Ajouter dans le header :

```
<link href="css/landing-campagne.css" rel="stylesheet">
```

Pour les shortcodes [lc-hero], [lc-benefits], [lc-testimonials], etc.

Étape 6+ : Ajouts progressifs selon les besoins

À définir au fur et à mesure :
- Composants spécifiques
- Responsive
- Etc.

Fichiers concernés

Fichier	Chemin
Template	app/public/wp-content/themes/opquast-v2/page-landing-opquast.php
Header	app/public/wp-content/themes/opquast-v2/header-landing-opquast.php
Footer	app/public/wp-content/themes/opquast-v2/footer-landing-opquast.php
CSS	app/public/wp-content/themes/opquast-v2/css/page-landing-opquast.css

---

Vérification

Après chaque étape :
1. Rafraîchir la page test
2. Vérifier visuellement le rendu
3. Passer à l'étape suivante

Approche

- **Progressive** : une étape à la fois
- **Sémantique** : HTML5 propre, IDs simples
- **Réutilisation** : styles existants du site (_variables.css, landing-campagne.css)

Doter l'agent de « compétences » : les skills

Il est possible de préparer des commandes complexes mobilisables par les agents

Context

Agent Skills

Agent Skills is an open standard for extending AI agents with specialized capabilities. Skills package domain-specific knowledge and workflows that agents can use to perform specific tasks.

What are skills?

A skill is a portable, version-controlled package that teaches agents how to perform domain-specific tasks. Skills can include both instructions and executable scripts or code that agents can run.

Portable

Skills work across any agent that supports the Agent Skills standard.

Version-controlled

Skills are stored as files and can be tracked in your repository, or installed via GitHub repository links.

Executable

Skills can include scripts and code that agents execute to perform tasks.

Progressive

Skills load resources on demand, keeping context usage efficient.

CRÉER AVEC CLAUDE CODE

Étendre Claude avec des compétences

Copier la page

Créez, gérez et partagez des compétences pour étendre les capacités de Claude dans Claude Code. Inclut les commandes slash personnalisées.

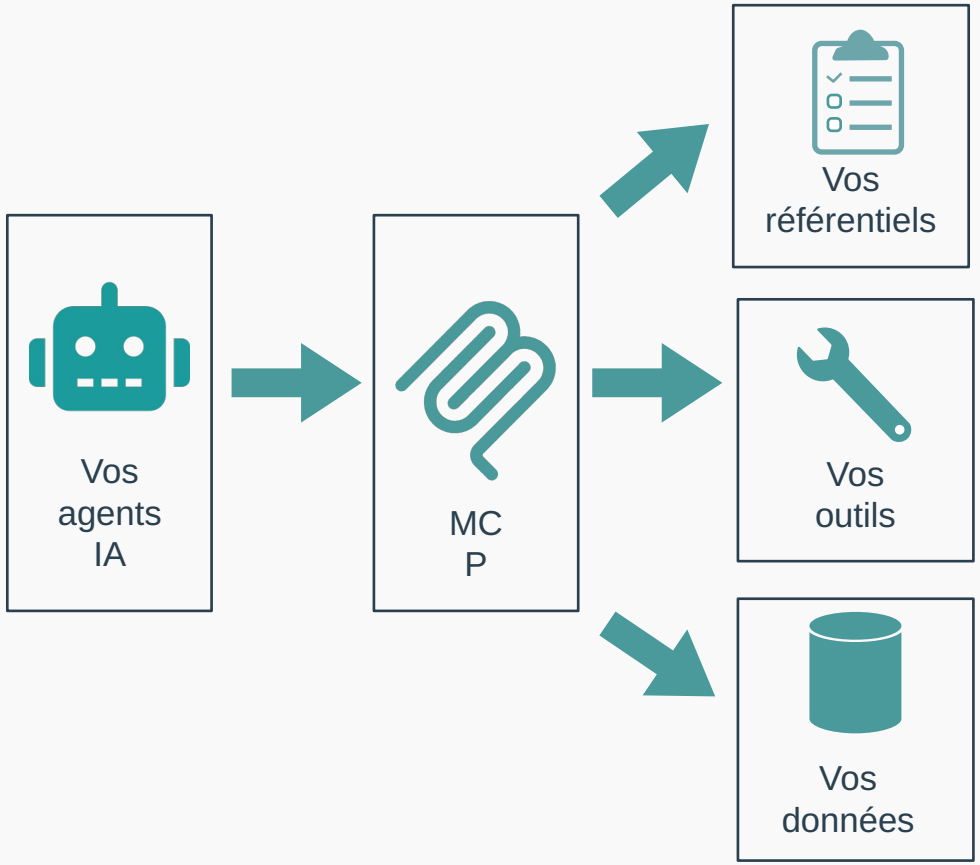
Les compétences étendent ce que Claude peut faire. Créez un fichier `SKILL.md` avec des instructions, et Claude l'ajoute à sa boîte à outils. Claude utilise les compétences quand c'est pertinent, ou vous pouvez en invoquer une directement avec `/skill-name`.

Pour les commandes intégrées comme `/help` et `/compact`, voir **mode interactif**.

Les commandes slash personnalisées ont été fusionnées dans les compétences. Un fichier à `.claude/commands/review.md` et une compétence à `.claude/skills/review/SKILL.md` créent tous les deux `/review` et fonctionnent de la même manière. Vos fichiers `.claude/commands/` existants continuent de fonctionner. Les compétences ajoutent des fonctionnalités optionnelles : un répertoire pour les fichiers de support, un frontmatter pour **contrôler si vous ou Claude les invoquez**, et la capacité pour Claude de les charger automatiquement quand c'est pertinent.

Les compétences Claude Code suivent la norme ouverte **Agent Skills**, qui fonctionne sur plusieurs outils d'IA. Claude Code étend la norme avec des fonctionnalités supplémentaires comme **le contrôle d'invocation**, **l'exécution de sous-agents**, et **l'injection de contexte dynamique**.

MCP — Model Context Protocol



Faire le lien avec l’intégration continue

Depuis un agent

Une fois un développement réalisé, l’agent IA appelle les tests avec les « hooks »

Ex : réalise une série de tests après chaque tâche

Depuis la fonderie logicielle

Lorsque les tests échouent, la fonderie appelle un agent IA pour la remédier.

Ex : propose des corrections pour que tout passe

Attention : lorsque les agents sont appelés “automatiquement” et qu’on utilise les APIs hors abonnements : les coûts explosent..

Attention. L’agent est fourbe.

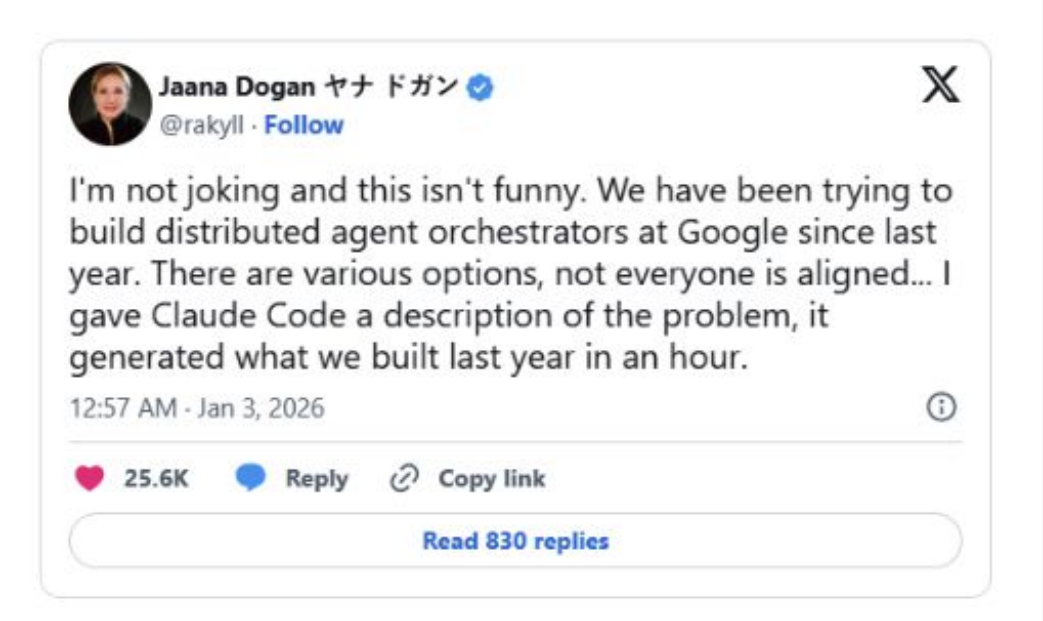
« Tu peux lui demander de tenir compte du référentiel Opquast. Il s’en tamponne. C’est Claude. Il sache. »
Laurent Denis

« Pour apporter des garanties, il faut combiner les agents IA (fonctionnement probabiliste) avec des scripts classiques (fonctionnement déterministe) »
Benjamin Hannache

Vous avez raison. Les regles 127+ que j'ai extraites lors du second fetch sont probablement des approximations generees par le modele IA, pas les vraies regles du site. La page est trop longue et se tronque. Laissez-moi essayer de recuperer les vraies regles de maniere fiable.

Le fichier d'audit est completement faux a partir de la regle 127. Les vrais numeros et contenus des regles sont differents. Je regenere l'integralite du rapport.

IA dans l’ingénierie : ca bouge très vite !



Au programme

1. Introduction - vue d’ensemble
2. MCP Opquast : présentation
3. MCP Opquast : retour d’XP (📣)
4. Roadmap & Conclusion

Présentation du périmètre

4. Outils disponibles

Le serveur MCP Opquast expose 7 outils :

Outil	Description
`get_rule`	Récupérer une règle par son numéro (1–245). Formats : résumé, détaillé ou actionnable.
`search_rules`	Rechercher des règles par mot-clé, avec filtrage optionnel par thème ou tag.
`get_rules_by_theme`	Obtenir toutes les règles d'un thème donné.
`list_themes`	Lister les 14 thèmes disponibles avec le nombre de règles par thème.
`list_versions`	Lister les versions du référentiel disponibles.
`get_audit_checklist`	Obtenir le protocole d'audit structuré pour une URL.
`generate_audit_report`	Générer un rapport d'audit HTML ou JSON avec synthèse et recommandations.

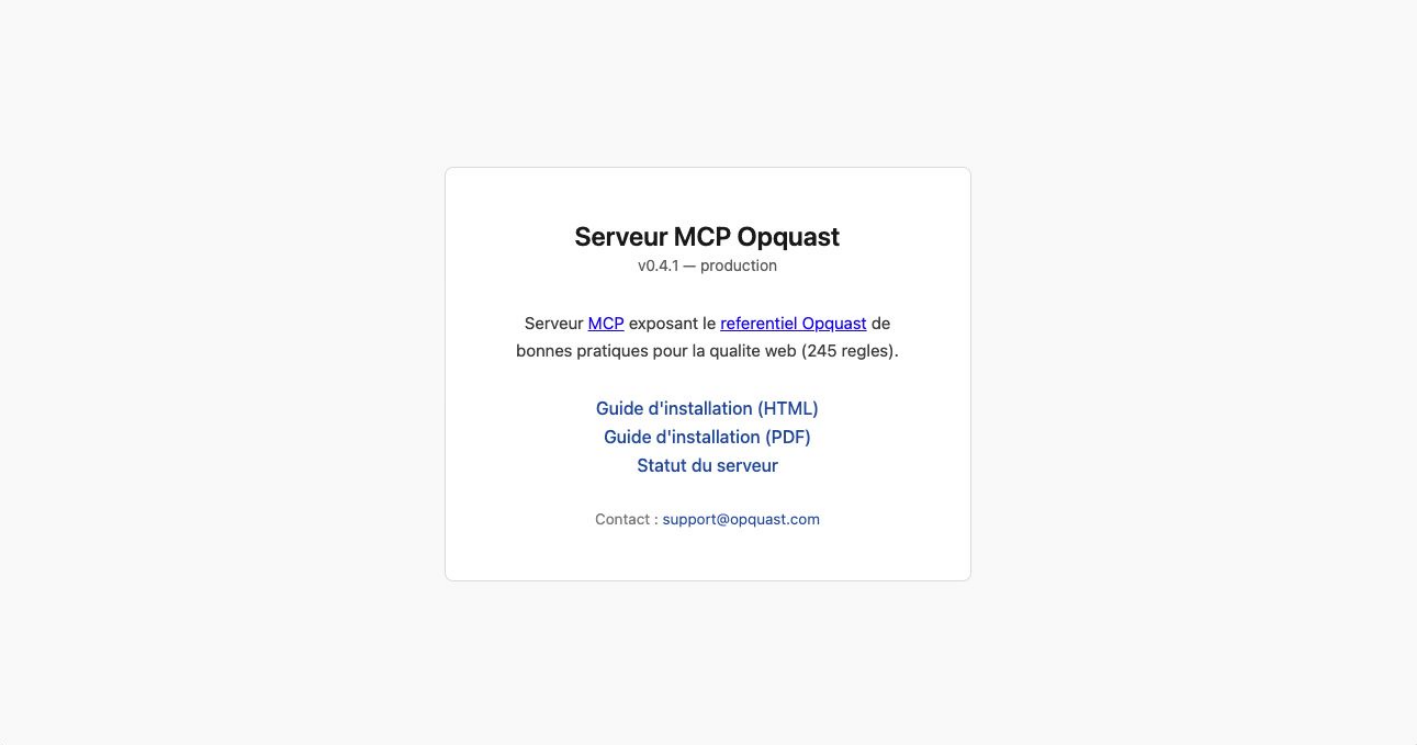
En bref :

Consulter le référentiel (pour une éventuelle exploitation IA pour auditer / améliorer)

Générer un rapport HTML

Installation

mcp.opquast.com



Installation

1. Prérequis

Avant de commencer, assurez-vous de disposer des éléments suivants :

☐ **Claude Code CLI** — npm install -g @anthropic-ai/claude-code
`https://docs.anthropic.com/en/docs/claude-code`

☐ **Node.js 18+** — Pour l'outil d'inspection des pages (Playwright ou Chrome DevTools). Recommandé : v20 ou v22.
`https://nodejs.org`

☐ **Système compatible** — macOS 14+, Windows 11+, ou Linux (Debian 12+, Ubuntu 22.04+). Requis pour Playwright. Sur Linux, le script installe automatiquement les dépendances système.

☐ **Google Chrome** — Uniquement si vous choisissez Chrome DevTools (optionnel avec Playwright)
`https://www.google.com/chrome/`

Installation

2. Installation

Installation automatique (recommandé)

Exécutez cette commande dans votre terminal en remplaçant `<TOKEN>` par votre token :

```
# Installation globale (disponible dans tous vos projets)
curl -sSL https://mcp.opquast.com/install.sh | bash -s -- <TOKEN>

# Ou installation locale (projet courant uniquement)
curl -sSL https://mcp.opquast.com/install.sh | bash -s -- <TOKEN> --local
```

Ce script configure automatiquement :

- Le serveur **Opquast MCP** (accès aux 245 règles du référentiel)
- Un **outil d'inspection des pages** au choix :
  - **Playwright** (recommandé) — installe son propre navigateur Chromium
  - **Chrome DevTools** — utilise Google Chrome installé sur la machine
- Les **autorisations d'outils** (pré-approbation des appels MCP, avec votre accord)
- L'envoi optionnel de **statistiques anonymes** (scope, outil choisi, autorisations — aucune donnée personnelle)

> **Autorisations :** Lors de l'installation, le script vous propose de pré-autoriser les outils MCP. Si vous acceptez, un fichier `.claude/settings.json` est créé dans le répertoire courant avec les permissions appropriées. Les appels aux outils seront alors automatiquement approuvés dans ce projet. Vous pouvez aussi utiliser les flags `--authorize`, `--stats`, `--playwright` ou `--chrome-devtools` pour un mode non-interactif.

> **Linux :** Si vous choisissez Playwright, le script installe automatiquement les dépendances système nécessaires (`--with-deps`). Cette étape peut demander des droits administrateur (`sudo`).

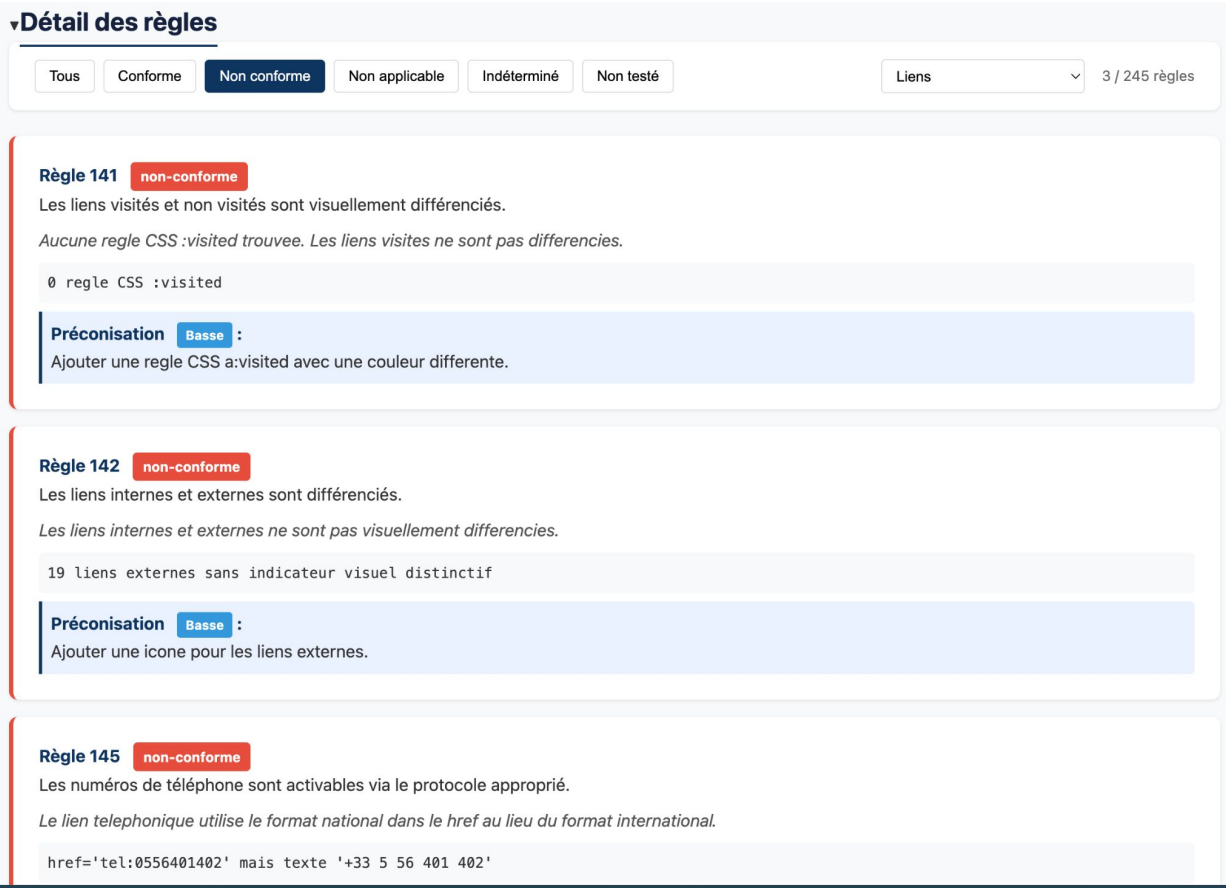
Les résultats

Synthèse

83.8%	98	19	55	6	67
CONFORMITÉ	CONFORMES	NON CONFORMES	NON APPLICABLES	INDÉTERMINÉS	NON TESTÉES

Conformité par thème

Les résultats



Au programme

1. Introduction - vue d’ensemble
2. MCP Opquast : présentation
3. MCP Opquast : retour d’XP (📣)
4. Roadmap & Conclusion

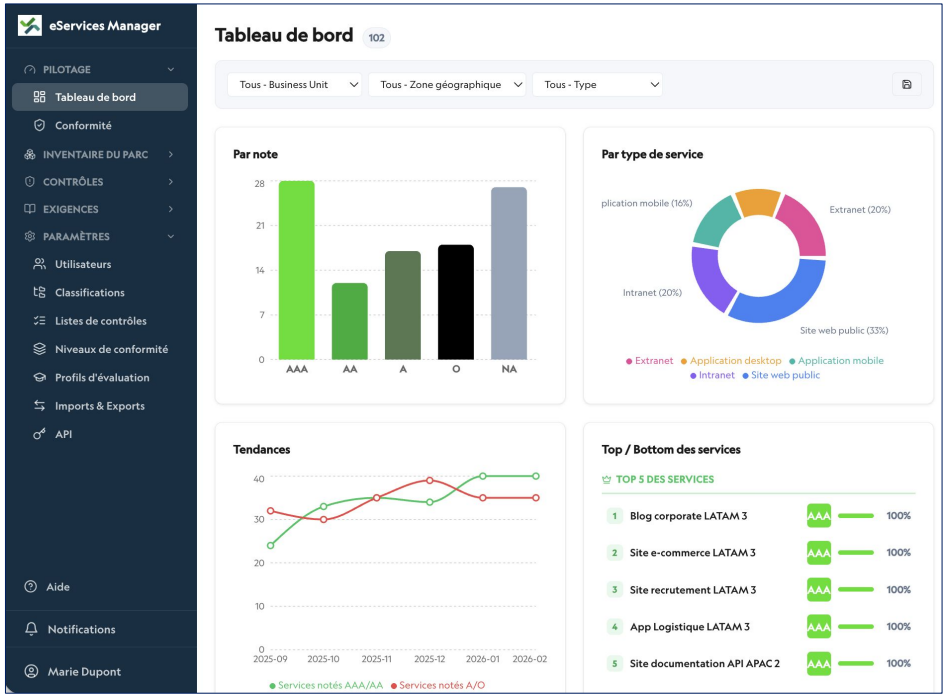
Opquast eServices Manager

Opquast eServices Manager est une plateforme de gestion de la qualité & conformité des services numériques (sites web, intranets, applications mobiles et desktop).

Elle permet aux organisations de :

- Centraliser l'inventaire de leurs services numériques
- Définir des référentiels de contrôles adaptés à chaque domaine (accessibilité, sécurité, RGPD, éco conception, charte graphique...)
- Réaliser des audits structurés et tracer l'historique des évaluations
- Programmer la réalisation de contrôles automatisés
- Attribuer des notes de conformité (personnalisables) et suivre leur évolution
- Piloter la remédiation et produire des reportings pour le management

Sera disponible en version Open Source et Commerciale



Un retour d'expérience

🖥️ Application auditée

eServices Manager — outil SaaS interne de pilotage de la conformité numérique

React 18 + Vite + TypeScript
Node.js + Express + Prisma
PostgreSQL, Clever Cloud

📋 Référentiel Opquast

245 règles de qualité numérique
14 thèmes (Structure, Sécurité, Formulaires, Navigation, etc.)

🔧 Dispositif MCP

Claude Opus 4 (1M context)
MCP Opquast — checklist & rapport
MCP Chrome DevTools — inspection page DOM, réseau, screenshots

3 sous-agents lancés en parallèle

Workflow en 3 phases

Phase 1 — AUDIT
Inspection + évaluation des 245 règles

→

Phase 2 — PLAN
Analyse, faux positifs, plan en 5 phases

→

Phase 3 — FIX
Implémentation des corrections (14 fichiers)

Un retour d’expérience

1. Collecte préalable des données (agent principal)

Navigation dans l'app (login → dashboard → assets → legal) via Chrome DevTools MCP
Extraction : DOM complet, meta tags, headings, forms, tables, aria, HTTP headers, network requests, screenshots
Données transmises aux sous-agents comme contexte → évite les conflits d'accès au navigateur

2. Agent DOM

102 règles — 6 thèmes

Structure et code (15)
Images et médias (12)
Liens (17)
Navigation (20)
Formulaires (30)
Internationalisation (8)

⏱️ 4 min 32 s

2. Agent Réseau

34 règles — 2 thèmes

Serveur et performances (13)
Sécurité (21)

Inspection du code source backend pour vérifier les headers de prod

⏱️ 6 min 38 s

2. Agent Visuel

109 règles — 6 thèmes

Présentation (17)
Contenus (14)
Contact (17)
Données perso (15)
E-Commerce (39 → N/A)
Newsletter (7 → N/A)

⏱️ 4 min 47 s

💡 Les 3 agents tournent en parallèle (background) — temps total d'audit : ~7 min au lieu de ~16 min séquentiel

Un retour d’expérience

```
> peux tu faire une analyse opquast de l'application (fais un bilan, mais n'implémente rien)

● Je vais réaliser un audit Opquast de l'application. Commençons par identifier les outils disponibles et les thèmes Opquast.

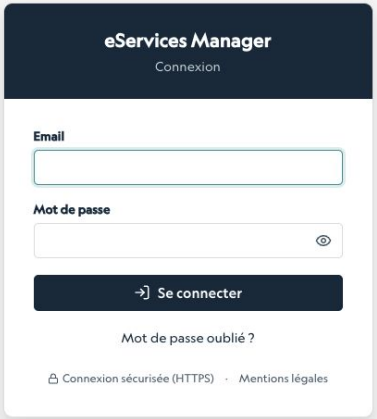
● J'utiliserai MCP chrome-devtools pour inspecter la page. Commençons par lister les thèmes Opquast et vérifier l'état du navigateur.

● opquast - list_themes (MCP)
  └ {
      "themes": [
        {
    … +59 lines (ctrl+o to expand)

● chrome-devtools - list_pages (MCP)
  └ ## Pages
    1: about:blank [selected]

● L'application n'est pas encore ouverte dans le navigateur. Laissez-moi vérifier l'URL locale de l'app.
```

Un retour d’expérience



Un retour d’expérience

```
● Connecté avec succès sur le tableau de bord. Je vais maintenant lancer l'audit Opquast complet. Je commence par collecter les données de la page, puis je
  délègue l'analyse par groupes de thèmes à des sous-agents.

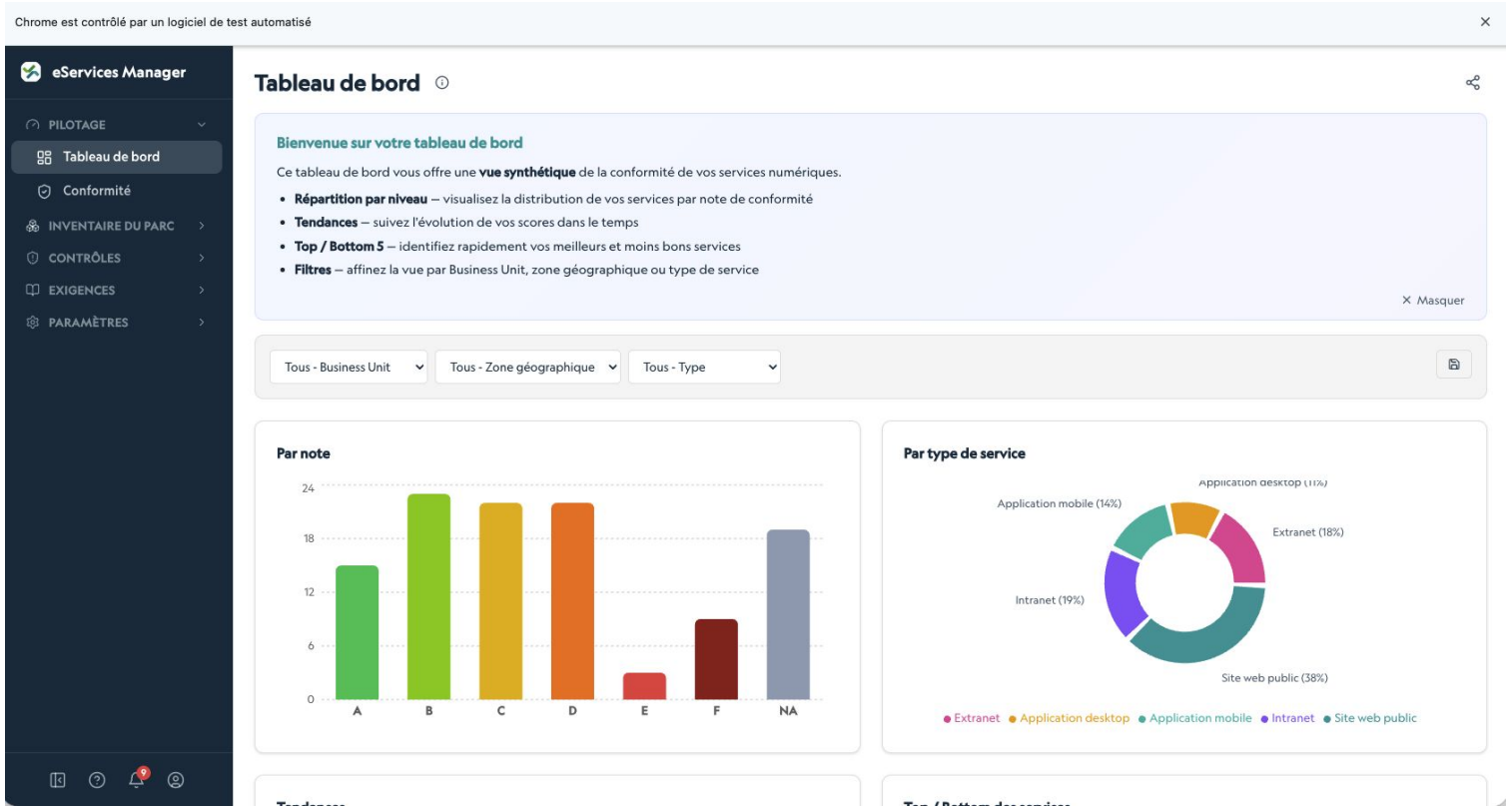
● Collectons les données HTML, réseau et captures des pages clés.

● chrome-devtools - evaluate_script (MCP)(function: "() => {\n  return {\n    title: document.title,\n    lang: document.documentElement.lang,\n    charset:
                                         document.characterSet,\n    doctype: document.doctype ? document.doctype.name : null,\n    metaTags:
                                         Array.from(document.querySelectorAll('meta')).map(m => ({\n      name: m.getAttribute('name'),\n      property:
                                         m.getAttribute('property'),\n      content: m.getAttribute('content'),\n      httpEquiv: m.getAttribute('http-equiv'),\n
                                              charset: m.getAttribute('charset')\n    })),\n    headings:
                                         Array.from(document.querySelectorAll('h1,h2,h3,h4,h5,h6')).map(h => ({\n      level: h.tagName,\n      text:
                                         h.textContent?.trim()\n    })),\n    images: Array.from(document.querySelectorAll('img')).map(img => ({\n      src:
                                         img.src,\n      alt: img.alt,\n      hasAlt: img.hasAttribute('alt'),\n      width: img.width,\n      height:
                                         img.height\n    })),\n    links: Array.from(document.querySelectorAll('a')).map(a => ({\n      href: a.href,\n      text:
                                          a.textContent?.trim(),\n      target: a.target,\n      rel: a.rel,\n      ariaLabel: a.getAttribute('aria-label')\n
                                         })).slice(0, 50),\n    forms: Array.from(document.querySelectorAll('form')).map(f => ({\n      action: f.action,\n
                                         method: f.method,\n      inputs: Array.from(f.querySelectorAll('input,select,textarea')).map(i => ({\n        type:
                                         i.type,\n        name: i.name,\n        id: i.id,\n        hasLabel:
                                         !!document.querySelector(`label[for=\"${i.id}\"]`),\n        required: i.required,\n        placeholder: i.placeholder,\n
                                                 ariaLabel: i.getAttribute('aria-label')\n      }))\n    })),\n    landmarks: {\n      header:
                                         document.querySelectorAll('header').length,\n      nav: document.querySelectorAll('nav').length,\n      main:
                                         document.querySelectorAll('main').length,\n      footer: document.querySelectorAll('footer').length,\n      aside:
                                         document.querySelectorAll('aside').length\n    },\n    skipLink: !!document.querySelector('a[href=\"#main-content\"],
                                         a[href=\"#content\"]'),\n    viewport: document.querySelector('meta[name=\"viewport\"]')?.content,\n    favicon:
                                         !!document.querySelector('link[rel=\"icon\"], link[rel=\"shortcut icon\"]')\n  }\n}")
  └ Script ran on page and returned:
    ```json
    {
    … +91 lines (ctrl+o to expand)

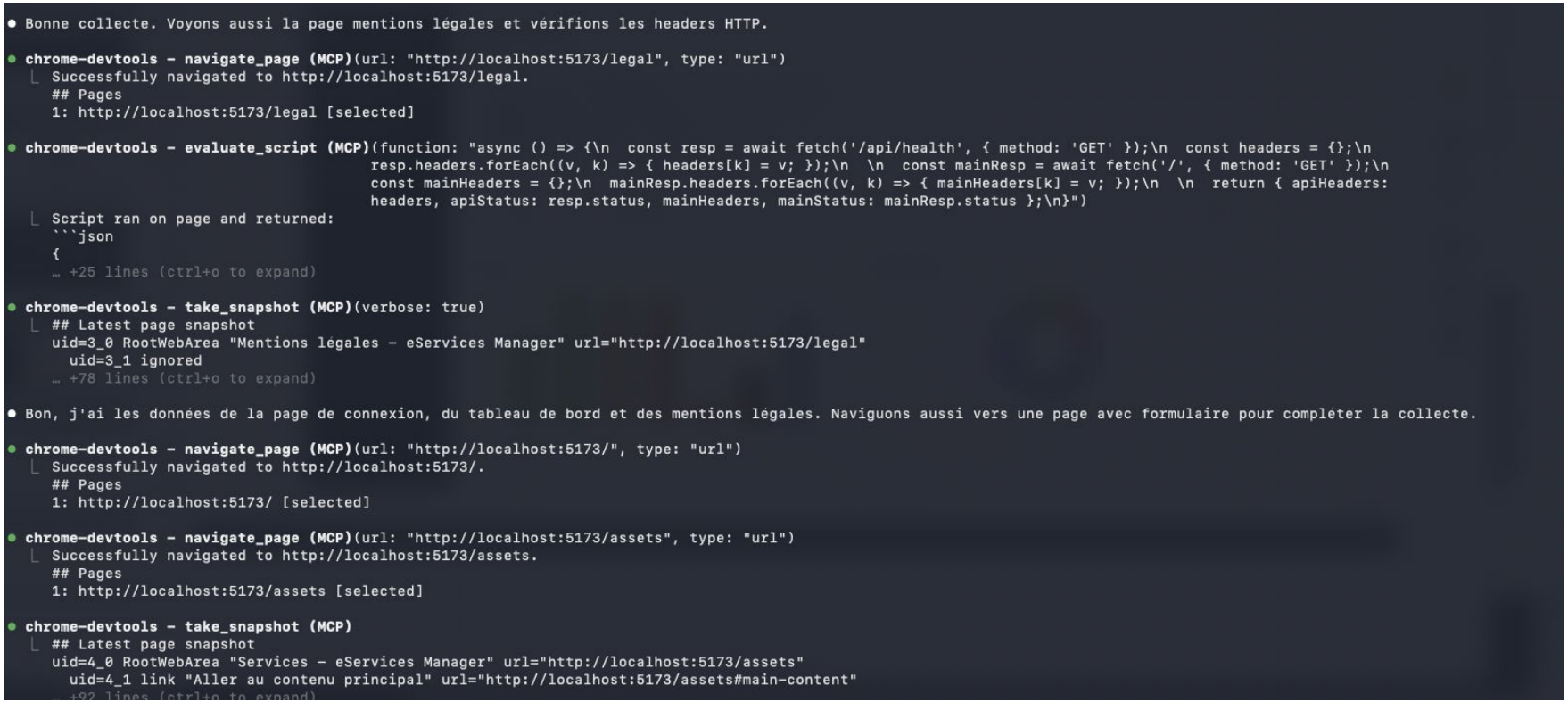
● chrome-devtools - take_screenshot (MCP)
  └ Took a screenshot of the current page's viewport.
  └ [Image]

● chrome-devtools - list_network_requests (MCP)(resourceTypes: ["document","stylesheet","script","font","image"])
  └ ## Network requests
    Showing 1-108 of 108 (Page 1 of 1).
    reqid=1 GET http://localhost:5173/ [200]
    … +51 lines (ctrl+o to expand)
```
```

Un retour d’expérience



Un retour d'expérience



Un retour d’expérience

```
● Bonne collecte. Voyons aussi la page mentions légales et vérifions les headers HTTP.

● chrome-devtools - navigate_page (MCP)(url: "http://localhost:5173/legal", type: "url")
  └ Successfully navigated to http://localhost:5173/legal.
    ## Pages
    1: http://localhost:5173/legal [selected]

● chrome-devtools - evaluate_script (MCP)(function: "async () => {\n  const resp = await fetch('/api/health', { method: 'GET' });\n  const headers = {};\n
                                         resp.headers.forEach((v, k) => { headers[k] = v; });\n  \n  const mainResp = await fetch('/', { method: 'GET' });\n
                                         const mainHeaders = {};\n  mainResp.headers.forEach((v, k) => { mainHeaders[k] = v; });\n  \n  return { apiHeaders:
                                         headers, apiStatus: resp.status, mainHeaders, mainStatus: mainResp.status };\n}")
  └ Script ran on page and returned:
    ```json
    {
    … +25 lines (ctrl+o to expand)

● chrome-devtools - take_snapshot (MCP)(verbose: true)
  └ ## Latest page snapshot
    uid=3_0 RootWebArea "Mentions légales - eServices Manager" url="http://localhost:5173/legal"
      uid=3_1 ignored
    … +78 lines (ctrl+o to expand)

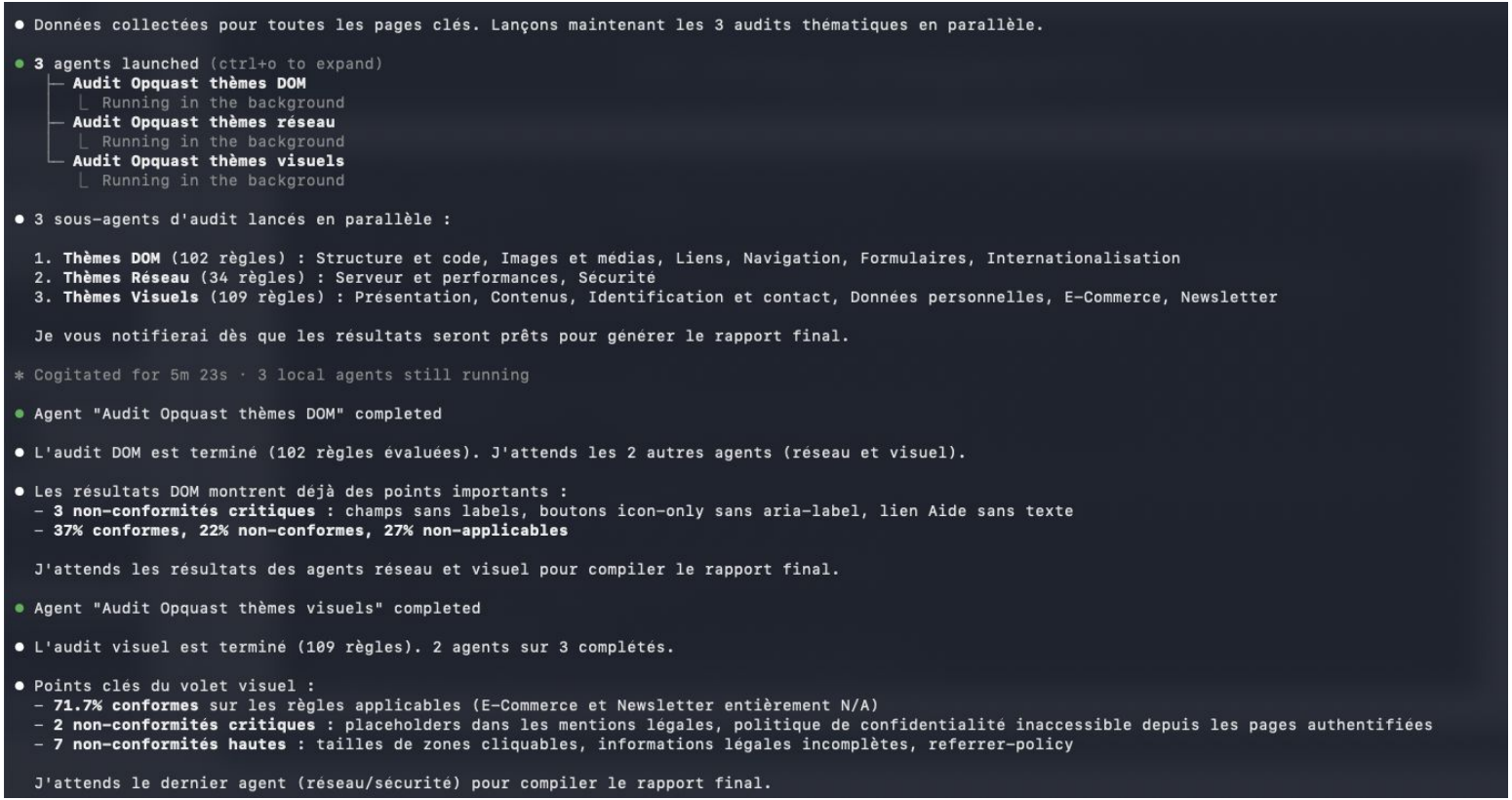
● Bon, j'ai les données de la page de connexion, du tableau de bord et des mentions légales. Naviguons aussi vers une page avec formulaire pour compléter la collecte.

● chrome-devtools - navigate_page (MCP)(url: "http://localhost:5173/", type: "url")
  └ Successfully navigated to http://localhost:5173/.
    ## Pages
    1: http://localhost:5173/ [selected]

● chrome-devtools - navigate_page (MCP)(url: "http://localhost:5173/assets", type: "url")
  └ Successfully navigated to http://localhost:5173/assets.
    ## Pages
    1: http://localhost:5173/assets [selected]

● chrome-devtools - take_snapshot (MCP)
  └ ## Latest page snapshot
    uid=4_0 RootWebArea "Services - eServices Manager" url="http://localhost:5173/assets"
      uid=4_1 link "Aller au contenu principal" url="http://localhost:5173/assets#main-content"
    … +92 lines (ctrl+o to expand)
```
```

Un retour d'expérience



```
● Données collectées pour toutes les pages clés. Lançons maintenant les 3 audits thématiques en parallèle.

● 3 agents launched (ctrl+o to expand)
   ├─ Audit Opquast thèmes DOM
   │  └ Running in the background
   ├─ Audit Opquast thèmes réseau
   │  └ Running in the background
   └─ Audit Opquast thèmes visuels
      └ Running in the background

● 3 sous-agents d'audit lancés en parallèle :

  1. Thèmes DOM (102 règles) : Structure et code, Images et médias, Liens, Navigation, Formulaires, Internationalisation
  2. Thèmes Réseau (34 règles) : Serveur et performances, Sécurité
  3. Thèmes Visuels (109 règles) : Présentation, Contenus, Identification et contact, Données personnelles, E-Commerce, Newsletter

  Je vous notifierai dès que les résultats seront prêts pour générer le rapport final.

✻ Cogitated for 5m 23s · 3 local agents still running

● Agent "Audit Opquast thèmes DOM" completed

● L'audit DOM est terminé (102 règles évaluées). J'attends les 2 autres agents (réseau et visuel).

● Les résultats DOM montrent déjà des points importants :
  - 3 non-conformités critiques : champs sans labels, boutons icon-only sans aria-label, lien Aide sans texte
  - 37% conformes, 22% non-conformes, 27% non-applicables

  J'attends les résultats des agents réseau et visuel pour compiler le rapport final.

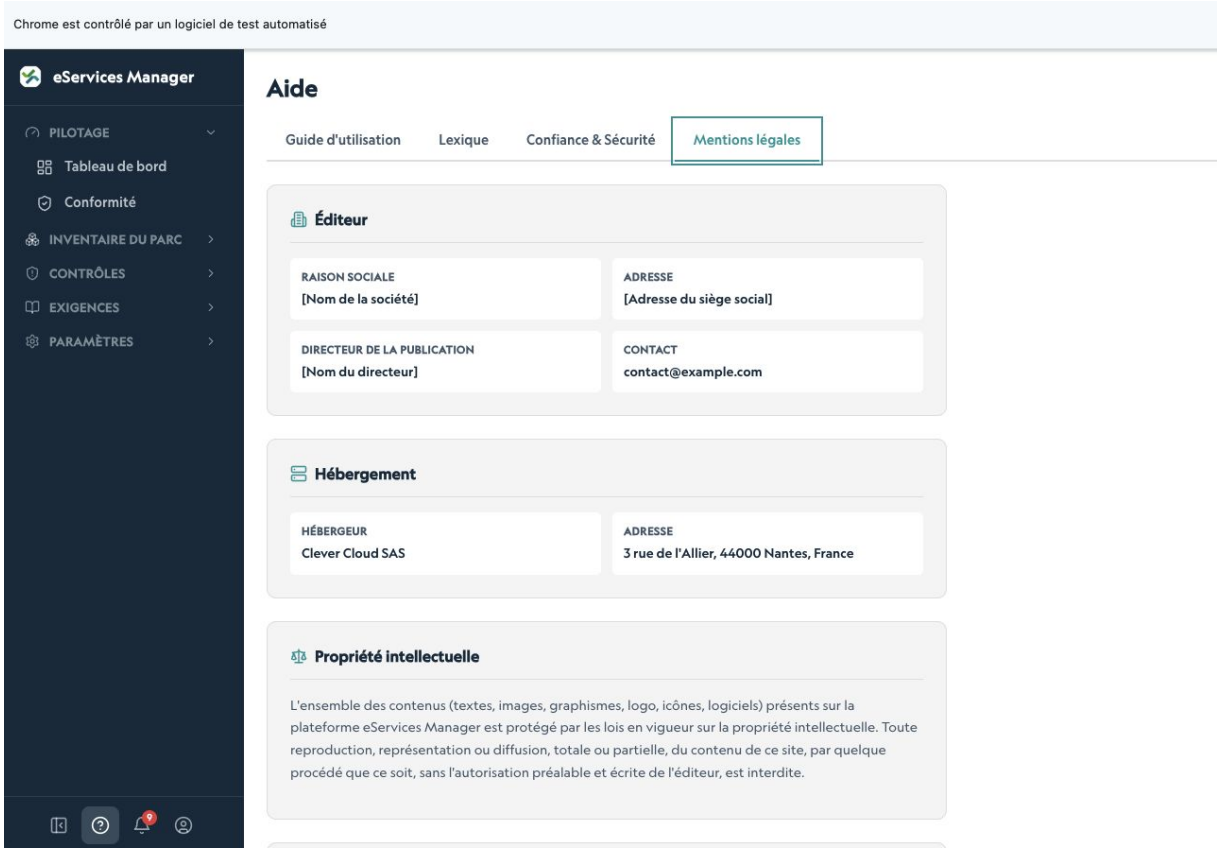
● Agent "Audit Opquast thèmes visuels" completed

● L'audit visuel est terminé (109 règles). 2 agents sur 3 complétés.

● Points clés du volet visuel :
  - 71.7% conformes sur les règles applicables (E-Commerce et Newsletter entièrement N/A)
  - 2 non-conformités critiques : placeholders dans les mentions légales, politique de confidentialité inaccessible depuis les pages authentifiées
  - 7 non-conformités hautes : tailles de zones cliquables, informations légales incomplètes, referrer-policy

  J'attends le dernier agent (réseau/sécurité) pour compiler le rapport final.
```

Un retour d’expérience



3

Phase Audit : résultats

Score global et répartition des 245 règles

69,5%
conformité

Conforme	82	33,5%
Non-conforme	36	14,7%
Non-applicable	90	36,7%
Indéterminé	12	4,9%
Non-testé	25	10,2%

Top 5 non-conformités

CRITIQUE	R69 — 27 champs sans label/aria-label
CRITIQUE	R159 — 20 boutons icon-only sans texte
CRITIQUE	R136 — Lien Aide avec texte vide
CRITIQUE	R101 — Mentions légales = placeholders
HAUTE	R234 — Hiérarchie titres H1→H3 (saute H2)

Conformité par thème (règles applicables)

Présentation 93% (14C / 1NC)	**Sécurité** 86% (12C / 2NC)	**Liens** 75% (6C / 2NC)	**Images** 80% (4C / 1NC)	**Contenus** 75% (6C / 2NC)	**Navigation** 62% (8C / 3NC)
Serveur & perf. 71% (5C / 2NC)	**Structure** 67% (8C / 4NC)	**Formulaires** 38% (3C / 5NC)	**Données perso** 80% (8C / 2NC)	**Contact** 42% (5C / 7NC)	**i18n** 25% (1C / 3NC)

5

Phase Plan : le plan d'implémentation

27 vraies non-conformités organisées en 5 phases

Phase 1
Accessibilité ARIA
~40 éléments : aria-labels sur inputs, selects, boutons, checkboxes, liens

Phase 2
Tableaux & titres
div→h2, scope=col, <caption> sur tables, hiérarchie H1→H2

Phase 3
Mentions légales
Lien privacy en sidebar, SIRET + téléphone, placeholders → données

Phase 4
Structure HTML
nav ul/li, tailles cliquables 36px+, 404 dans Layout

Phase 5
Améliorations
Logo optimisé, sitemap, <time>, <abbr>, lang="en"

Points clés du plan

- Regroupement par fichier (pas par règle) → minimise le nombre de modifications
- Réutilisation des patterns existants (form-label, aria-label du login, Breadcrumb)
- Prérequis identifiés : classe .sr-only, clés i18n avant les modifications de composants
- Phase 3A bloquée : données légales réelles nécessaires (SIRET, adresse, contacts)
- Distinction dev vs prod explicite pour les headers de sécurité (HSTS, CSP)
- Score cible estimé : 82-85% (vs 69,5% actuel)

Un retour d’expérience

```
● Agent "Audit Opquast thèmes réseau" completed

● Les 3 audits sont terminés. Compilons le rapport final.

● Récupérons les résultats détaillés de l'agent visuel pour le rapport combiné.

  Searched for 1 pattern (ctrl+o to expand)

● Bash(cat /Users/bhannache/.claude/projects/-Users-bhannache-Desktop-eService-Manager/9f7c1ae5-a0ec-490c-bce6-f8040b9cd852/tool-results/b2506zkit.txt | python3 -c "
      i…)
    └ Found 0 rules
      []

  Read 1 file (ctrl+o to expand)

● Bash(python3 << 'PYEOF'
      import json…)
    └ Found 109 rules from visual agent
      Saved to /tmp/visual_results.json

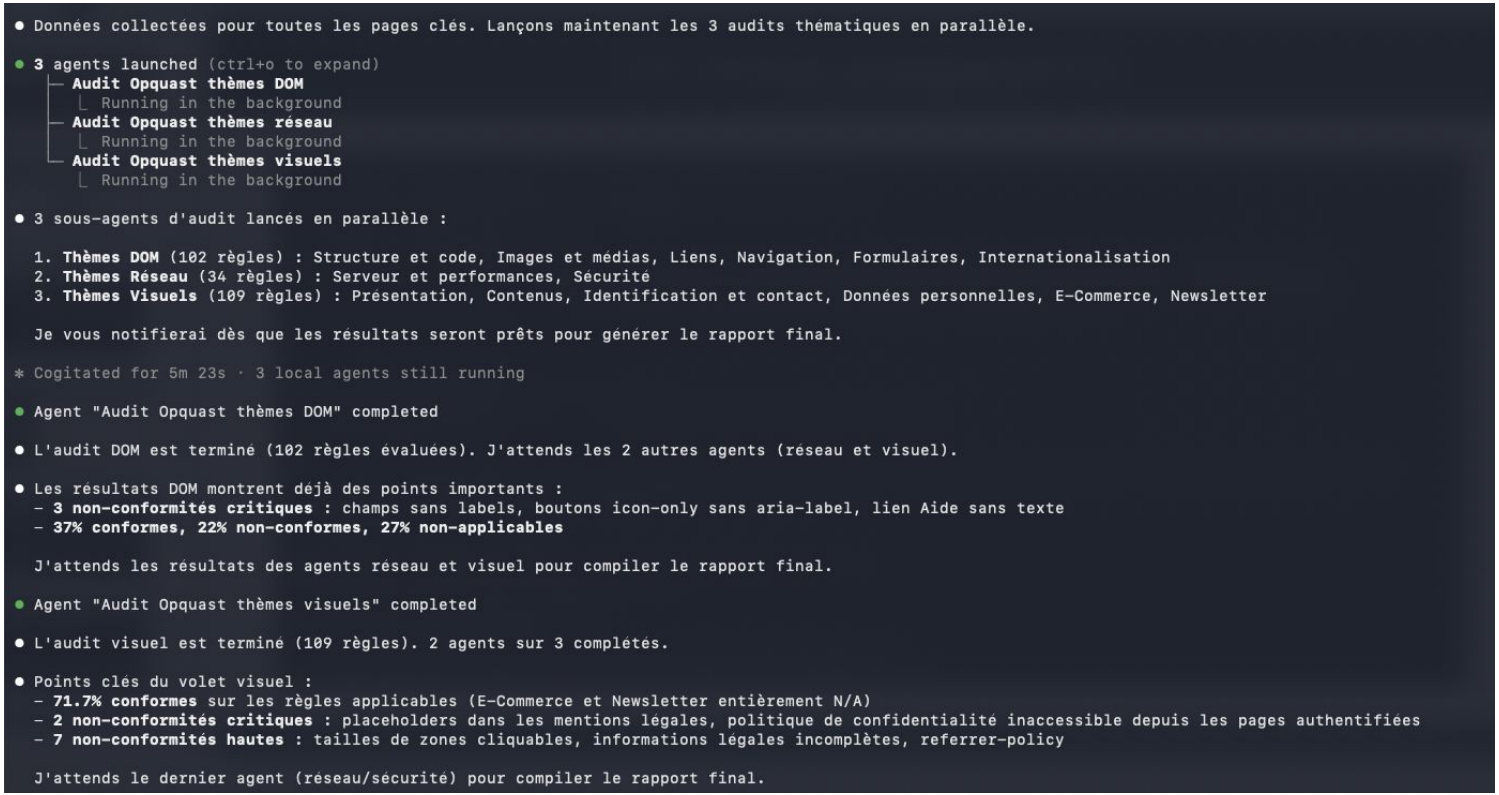
● Maintenant, combinons les 245 résultats et générons le rapport final.

● Bash(python3 << 'PYEOF'
      import json…)
    └ Total results: 245
      Unique rule numbers: 245

      Statuts: {'non-conforme': 36, 'conforme': 79, 'non-applicable': 93, 'indéterminé': 12, 'non-testé': 25}

● 245 règles, 0 doublons. Générons le rapport HTML final.
```

Un retour d'expérience



```
● Données collectées pour toutes les pages clés. Lançons maintenant les 3 audits thématiques en parallèle.

● 3 agents launched (ctrl+o to expand)
   ├─ Audit Opquast thèmes DOM
   │  └ Running in the background
   ├─ Audit Opquast thèmes réseau
   │  └ Running in the background
   └─ Audit Opquast thèmes visuels
      └ Running in the background

● 3 sous-agents d'audit lancés en parallèle :

  1. Thèmes DOM (102 règles) : Structure et code, Images et médias, Liens, Navigation, Formulaires, Internationalisation
  2. Thèmes Réseau (34 règles) : Serveur et performances, Sécurité
  3. Thèmes Visuels (109 règles) : Présentation, Contenus, Identification et contact, Données personnelles, E-Commerce, Newsletter

  Je vous notifierai dès que les résultats seront prêts pour générer le rapport final.

✻ Cogitated for 5m 23s · 3 local agents still running

● Agent "Audit Opquast thèmes DOM" completed

● L'audit DOM est terminé (102 règles évaluées). J'attends les 2 autres agents (réseau et visuel).

● Les résultats DOM montrent déjà des points importants :
  - 3 non-conformités critiques : champs sans labels, boutons icon-only sans aria-label, lien Aide sans texte
  - 37% conformes, 22% non-conformes, 27% non-applicables

  J'attends les résultats des agents réseau et visuel pour compiler le rapport final.

● Agent "Audit Opquast thèmes visuels" completed

● L'audit visuel est terminé (109 règles). 2 agents sur 3 complétés.

● Points clés du volet visuel :
  - 71.7% conformes sur les règles applicables (E-Commerce et Newsletter entièrement N/A)
  - 2 non-conformités critiques : placeholders dans les mentions légales, politique de confidentialité inaccessible depuis les pages authentifiées
  - 7 non-conformités hautes : tailles de zones cliquables, informations légales incomplètes, referrer-policy

  J'attends le dernier agent (réseau/sécurité) pour compiler le rapport final.
```

Un retour d’expérience

> *Le taux de conformité (69.5%) est calculé sur les 118 règles applicables et testées (82 conformes / 118).*

Répartition par thème

Thème	C	NC	N/A	Ind.	NT
Structure et code (15)	8	4	3	0	0
Images et médias (12)	4	1	7	0	0
Liens (17)	6	2	8	1	0
Navigation (20)	8	3	2	2	5
Formulaires (30)	3	5	10	0	12
Internationalisation (8)	1	3	3	0	1
Présentation (17)	14	1	0	1	1
Contenus (14)	6	2	4	1	1
Identification & contact (17)	5	7	2	0	3
Données personnelles (15)	8	2	0	0	5
Serveur & performances (13)	5	2	2	4	0
Sécurité (21)	12	2	2	4	1
E-Commerce (39)	0	0	39	0	0
Newsletter (7)	0	0	7	0	0

Un retour d’expérience

Non-conformités critiques (5)

#	Règle	Problème
1	**R69**	**22 champs + 5 selects sans label/aria-label** (checkboxes tableau, filtres, recherche)
2	**R159**	**20 boutons icon-only sans texte ni aria-label** (sidebar, notifications, actions)
3	**R136**	**Lien "Aide" avec texte vide** dans la sidebar
4	**R101**	**Mentions légales = placeholders** ([Nom de la société], contact@example.com...)
5	**R15**	**Politique de confidentialité inaccessible** depuis les pages authentifiées

Non-conformités hautes (12)

Règle	Problème
R234	Hiérarchie de titres : H1 → H3 (saute H2) sur le dashboard
R242	Pas de scope sur les <th> des tableaux
R243	Pas de <caption> sur les tableaux de données
R77	Selects de filtre sans étiquette visible
R137	Lien Aide sans libellé explicite
R186	Zones cliquables trop petites (< 44px)
R105-107	Coordonnées incomplètes (adresse, téléphone, SIRET)
R112	Pas de moyen de contacter le support
R114	Directeur de publication = placeholder
R199	Header HSTS absent (configuré en prod seulement)
R212	Content-Security-Policy absent en dev
R25	Referrer-Policy absent

Un retour d’expérience

Bilan Opquast – eServices Manager

Rapport HTML complet : https://mcp.opquast.com/reports/feed9a59960b4939924540d137a8b0ee (expire dans 1h)

Score global : 69.5% de conformité

Statut	Nombre	%
Conforme	82	33.5%
Non-conforme	36	14.7%
Non-applicable	90	36.7%
Indéterminé	12	4.9%
Non-testé	25	10.2%

Un retour d’expérience

Points forts

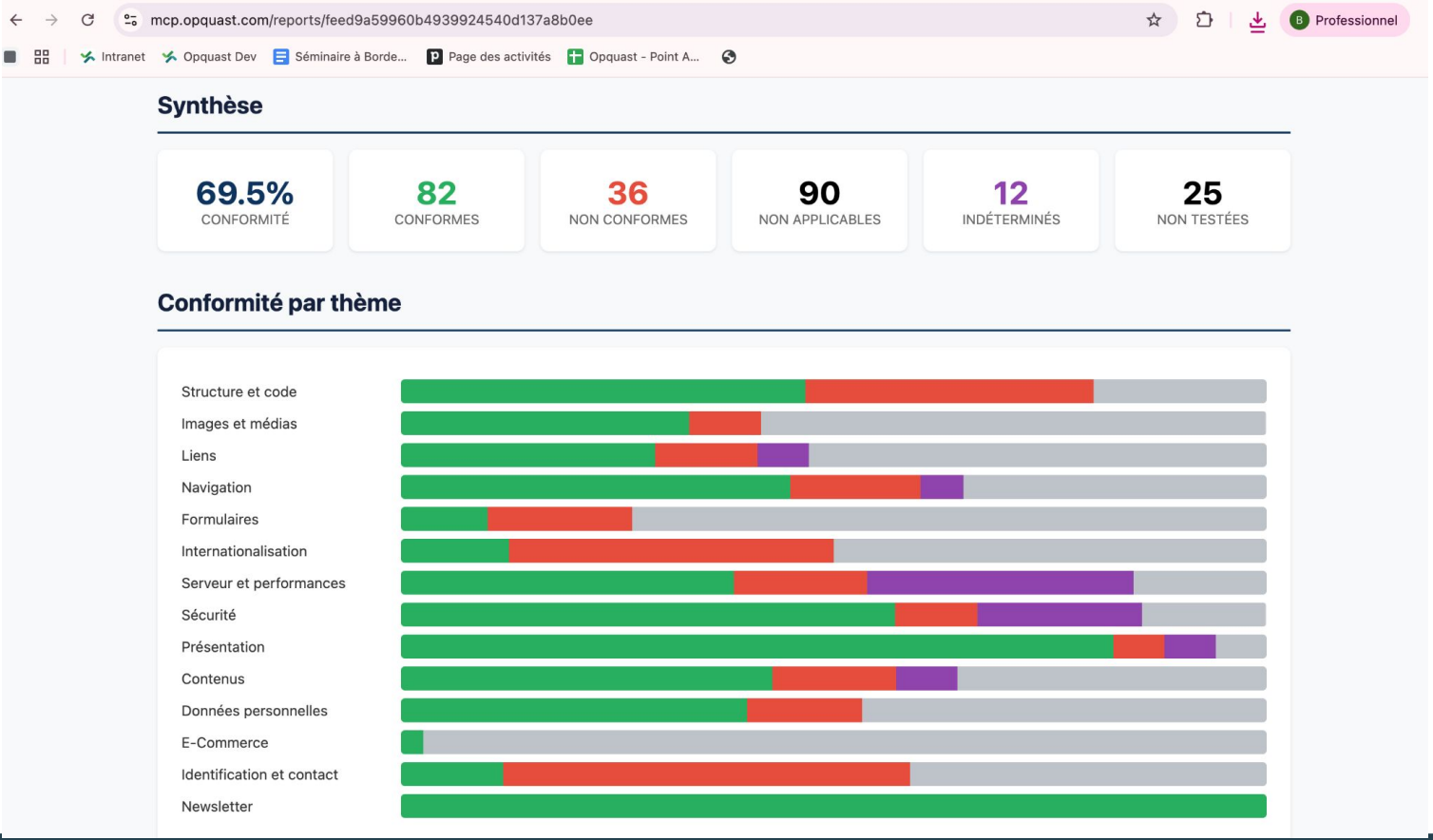
- **Sécurité** : bcrypt, gestion complète des mots de passe, X-Frame-Options, X-Content-Type-Options, pas de fuite X-Powered-By
- **Présentation** : charte graphique cohérente, bon contraste, responsive, zoom non bloqué
- **Navigation** : skip link, sidebar persistante, logo = lien accueil, aria-current sur items actifs
- **Contenus** : titres de page uniques, dates lisibles, pas de popup ni son auto-joué

Recommandations prioritaires

1. **Accessibilité formulaires** : ajouter `aria-label` sur tous les boutons icon-only et `<label>` sur les champs de formulaire
2. **Mentions légales** : remplacer tous les placeholders par les vraies informations (raison sociale, SIRET, adresse, contacts)
3. **Lien mentions légales/confidentialité** : ajouter un lien permanent dans le footer ou la sidebar accessible depuis toutes les pages
4. **Tableaux** : ajouter `scope="col"` et `<caption>` sur les tableaux de données
5. **Hiérarchie de titres** : corriger H3 → H2 sur le dashboard
6. **i18n** : ajouter un sélecteur de langue visible
7. **Sécurité** : vérifier HSTS et CSP en production, ajouter `preload` au HSTS

> *Note : L'audit a été réalisé en environnement de développement (localhost). Plusieurs points « indéterminés » (HTTPS, compression, minification, CSP, HSTS) sont probablement conformes en production. Un audit complémentaire sur l'URL de production est recommandé.*

Un retour d’expérience



4

Phase Plan : analyse critique des résultats

Détection des faux positifs et vérification dans le code source

Étape clé : ne pas faire confiance aveuglément aux résultats des agents

⚠️ 7 faux positifs détectés (sur 36 non-conformités)

R97 — autocomplete déjà présent sur le formulaire login
R25 — Referrer-Policy déjà configuré dans app.ts
R199 — HSTS configuré en production (invisible en dev)
R212 — CSP configuré en production (invisible en dev)
R133/R134 — Sélecteur de langue existant (préférences)
R156 — Composant Breadcrumb existant et utilisé

→ 19% des "non-conformités" étaient des faux positifs

🔍 Pourquoi ces faux positifs ?

Audit en environnement dev (localhost) :
- Headers de sécurité absents en dev mais configurés pour la production
- Vite dev server ≠ Express prod server

Inspection superficielle des agents :
- Agent n'a pas lu le code source du login pour vérifier autocomplete
- Sélecteur de langue dans un panneau caché (non visible dans le snapshot initial)

Méthode de vérification

- 2 agents Explore lancés en parallèle pour lire en profondeur les fichiers concernés (~15 fichiers clés)
- Croisement résultats d'audit × code source réel → chaque non-conformité vérifiée manuellement
- 1 agent Plan pour structurer l'implémentation avec le contexte vérifié

Un retour d'expérience

```
> propose moi un plan détaillé pour implémenter avant de démarrer

● Entered plan mode
  Claude is now exploring and designing an implementation approach.

● Je vais explorer le codebase pour comprendre les patterns existants avant de proposer le plan.

● 2 Explore agents finished (ctrl+o to expand)
   ├─ Explore accessibilité & formulaires · 31 tool uses · 89.0k tokens
   │  └ Done
   └─ Explore sécurité & mentions légales · 33 tool uses · 65.7k tokens
      └ Done

● Bonne exploration. Quelques découvertes importantes : le login a déjà autocomplete, le Referrer-Policy est déjà configuré, le breadcrumb existe
  déjà, et un sélecteur de langue est dans les préférences. Vérifions encore quelques fichiers clés.

  Searched for 1 pattern, read 1 file (ctrl+o to expand)

● Maintenant j'ai une vision complète. Plusieurs "non-conformités" de l'audit sont en fait des faux positifs :
  - R97 : autocomplete est déjà en place sur le login
  - R25 : Referrer-Policy est déjà configuré dans app.ts
  - R199/R212 : HSTS et CSP sont configurés en production
  - Le breadcrumb existe déjà et est utilisé sur 4 pages
  - Le sélecteur de langue existe dans le panneau de préférences

  Lançons un agent de planification pour structurer l'implémentation.

● Plan(Plan implémentation Opquast)
  └ Done (86 tool uses · 82.6k tokens · 7m 14s)
  (ctrl+o to expand)

● Bon plan. Vérifions quelques points clés avant de finaliser.

  Searched for 3 patterns, read 2 files (ctrl+o to expand)
```

Un retour d’expérience

● Le plan est organisé en **5 phases** par priorité décroissante :

1. **Phase 1 (Critique)** : Labels ARIA sur ~40 éléments — inputs, selects, boutons icon-only, lien Aide
2. **Phase 2 (Haute)** : Sémantique tableaux (scope, caption) + hiérarchie titres H1→H2
3. **Phase 3 (Haute/Critique)** : Mentions légales (placeholders → vraies données), lien confidentialité, contact support
4. **Phase 4 (Moyenne)** : Structure nav ul/li, tailles cliquables, 404 avec sidebar, lang="en", <abbr>
5. **Phase 5 (Basse)** : Logo optimisé, plan du site, métadonnées dates, mentions mineures

7 faux positifs écartés (autocomplete, Referrer-Policy, HSTS/CSP en prod, sélecteur de langue, breadcrumb).

Score projeté : ~82-85% après implémentation complète.

Point bloquant : Phase 3A nécessite les vraies données légales (raison sociale, SIRET, adresse, contacts).

Un retour d’expérience

Ordre d'exécution et dépendances

```
Phase 1A (sr-only CSS) ──┐
Phase 1B (clés i18n)   ──┼──> Phase 1C–1G (labels ARIA)
                         │
Phase 2 (tableaux, titres) ──> indépendant
Phase 3A (données légales) ──> bloqué par données client
Phase 3C–3D (liens)        ──> indépendant
Phase 4 (structure)        ──> indépendant
Phase 5 (mineur)           ──> dernière priorité
```

Fichiers critiques

Fichier	Phases	Type de changement
frontend/src/theme/index.css	1A, 4B	+sr-only, tailles min
frontend/src/i18n/fr.json + en.json	1B, 2C, 3A–B, 3D	+15 clés aria, placeholders, support
frontend/src/components/Layout.tsx	1E, 3C, 4A	aria-labels, lien legal, ul/li
frontend/src/pages/AssetsPage.tsx	1C, 2C	aria-labels, scope, caption
frontend/src/pages/DashboardPage.tsx	1D, 2A–B	aria-labels, h2, scope, caption
frontend/src/components/KebabMenu.tsx	1G	aria-label
frontend/src/components/PageGuideBanner.tsx	1F	aria-label
frontend/src/pages/LegalPage.tsx	3B, 4E	SIRET/tel, abbr
frontend/src/App.tsx	4C	route 404 dans Layout
frontend/src/pages/NotFoundPage.tsx	4C	prop embedded

6 Phase Correction : implémentation

14 fichiers modifiés, 175 ajouts, 62 suppressions

Fichiers modifiés

index.css	+sr-only, min-size 36px
fr.json + en.json	+18 clés aria, SIRET, téléphone
Layout.tsx	aria-labels, ul/li nav, lien legal
AssetsPage.tsx	aria-labels, scope, caption
DashboardPage.tsx	div→h2, aria-labels, scope
KebabMenu.tsx	aria-label + useTranslation
PageGuideBanner.tsx	aria-label sur toggle
ShareButton.tsx	aria-label
ExportButton.tsx	aria-label
FilterSaveLoad.tsx	aria-label
LegalPage.tsx + LegalContent.tsx	SIRET + téléphone
App.tsx	Route 404 dans Layout

Vérifications effectuées

✅ Contrôle qualité automatique

TypeScript : compilation sans erreur (tsc --noEmit)
Chrome DevTools : vérification aria-labels dans le DOM
Vérification visuelle : screenshot avant/après identique
Aucune régression visuelle détectée

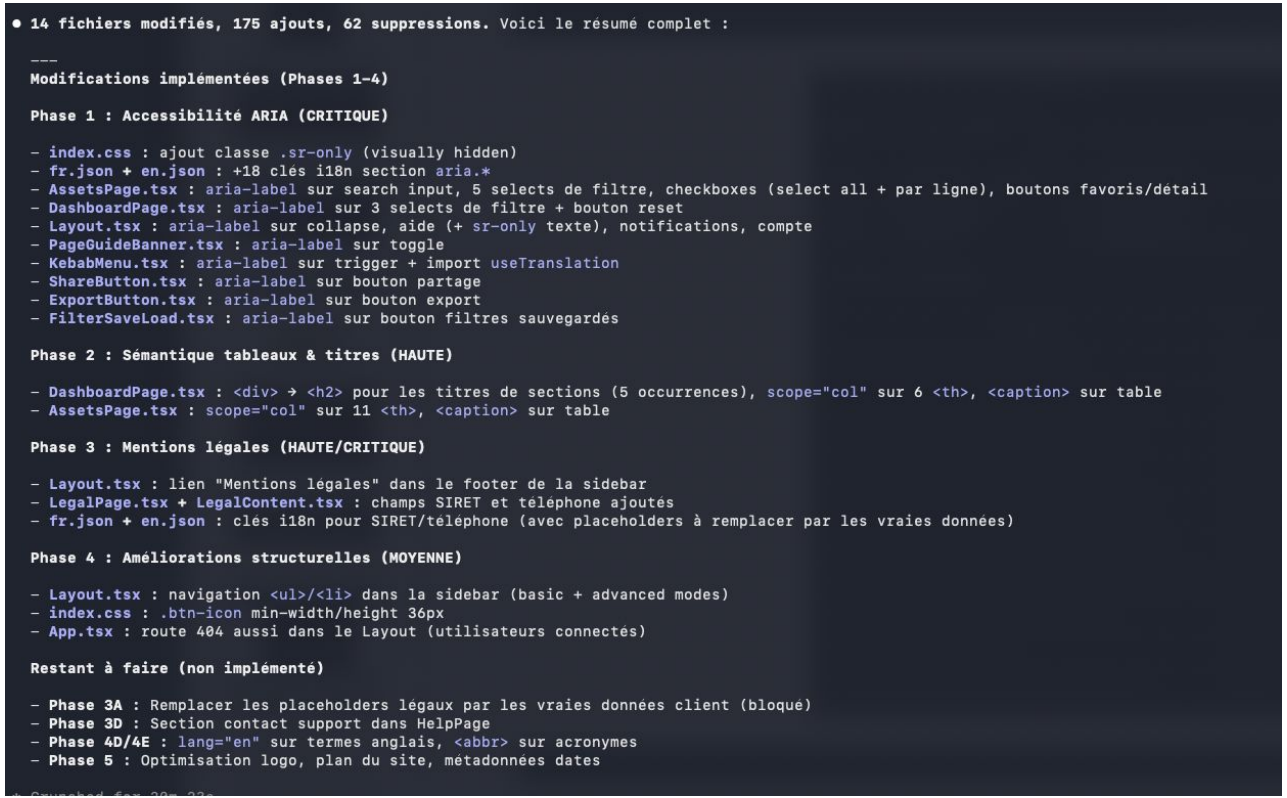
📊 Règles corrigées par phase

Phase 1 (ARIA) → R69, R77, R159, R136/R137 = 5 règles
Phase 2 (tables) → R234, R242, R243 = 3 règles
Phase 3 (legal) → R15, R105-R107 = 4 règles (partiel)
Phase 4 (struct.) → R235, R186, R225 = 3 règles

⏱️ Temps d'exécution

Audit complet : ~7 min (agents parallèles)
Planification : ~5 min (exploration + plan)
Implémentation : ~10 min (14 fichiers)
Total : ~22 min pour un audit + corrections

Un retour d'expérience



● **14 fichiers modifiés, 175 ajouts, 62 suppressions.** Voici le résumé complet :

Modifications implémentées (Phases 1–4)

Phase 1 : Accessibilité ARIA (CRITIQUE)

- **index.css** : ajout classe .sr-only (visually hidden)
- **fr.json + en.json** : +18 clés i18n section aria.*
- **AssetsPage.tsx** : aria-label sur search input, 5 selects de filtre, checkboxes (select all + par ligne), boutons favoris/détail
- **DashboardPage.tsx** : aria-label sur 3 selects de filtre + bouton reset
- **Layout.tsx** : aria-label sur collapse, aide (+ sr-only texte), notifications, compte
- **PageGuideBanner.tsx** : aria-label sur toggle
- **KebabMenu.tsx** : aria-label sur trigger + import useTranslation
- **ShareButton.tsx** : aria-label sur bouton partage
- **ExportButton.tsx** : aria-label sur bouton export
- **FilterSaveLoad.tsx** : aria-label sur bouton filtres sauvegardés

Phase 2 : Sémantique tableaux & titres (HAUTE)

- **DashboardPage.tsx** : → <h2> pour les titres de sections (5 occurrences), scope="col" sur 6 <th>, <caption> sur table
- **AssetsPage.tsx** : scope="col" sur 11 <th>, <caption> sur table

Phase 3 : Mentions légales (HAUTE/CRITIQUE)

- **Layout.tsx** : lien "Mentions légales" dans le footer de la sidebar
- **LegalPage.tsx + LegalContent.tsx** : champs SIRET et téléphone ajoutés
- **fr.json + en.json** : clés i18n pour SIRET/téléphone (avec placeholders à remplacer par les vraies données)

Phase 4 : Améliorations structurelles (MOYENNE)

- **Layout.tsx** : navigation / dans la sidebar (basic + advanced modes)
- **index.css** : .btn-icon min-width/height 36px
- **App.tsx** : route 404 aussi dans le Layout (utilisateurs connectés)

Restant à faire (non implémenté)

- **Phase 3A** : Remplacer les placeholders légaux par les vraies données client (bloqué)
- **Phase 3D** : Section contact support dans HelpPage
- **Phase 4D/4E** : lang="en" sur termes anglais, <abbr> sur acronymes
- **Phase 5** : Optimisation logo, plan du site, métadonnées dates

✻ Crunched for 20m 23s

Conclusion : 3 enseignements

1

MCP comme levier de productivité

L'orchestration de serveurs MCP spécialisés (Opquast + Chrome DevTools) permet un audit en quelques minutes. Le MCP est un protocole de composition, pas juste un outil isolé.

2

La vérification humaine reste indispensable

19% de faux positifs → l'IA sur-détecte. La phase de plan avec relecture (par l'IA) du code source est cruciale. Ne jamais appliquer aveuglément les résultats.

3

Le cycle audit → plan → fix en une session change la donne

Passer de "rapport PDF sur étagère" à "corrections mergées dans la branche" en 22 minutes. La qualité web devient un process itératif et rapide.

Au programme

1. Introduction - vue d’ensemble
2. MCP Opquast : présentation
3. MCP Opquast : retour d’XP (📣)
4. Roadmap & Conclusion

Roadmap

Amélioration du MCP

Estimer la fiabilité et les coûts via un banc de tests : panel de sites, panel de LLM

Optimisation : fonctionnement mixte probabiliste / déterministe

eService Manager

Phase pilote en cours : contactez-nous !

Sortie prochaine version commerciale et Open Source

Nouvelle édition du livre (Eyrolles 2026)

Qualité et conformité des services numériques

Trois nouveaux chapitres

IA et qualité numérique

Qualité logicielle

Normes, standards et réglementations

C’est à vous

Merci !

À vos questions