Eigenflow

GUIDE · CLAUDE CODE

Astryx, le design system open source de Meta

L'installer, changer de thème, brancher son serveur MCP à Claude Code, et notre vrai test sans puis avec, vérifiés le 15 novembre 2026.

PAR

Abyane Nabeebaccus

INSTAGRAM

@abyane_nb

ÉDITION

novembre 2026

Tu as commenté KIT, voici le guide. Astryx est le design system que Meta a rendu open source le 18 juin 2026 : des composants React, sept thèmes, et un serveur MCP qui donne sa documentation à Claude pendant qu'il code. Dans l'ordre : ce qu'il est et ce qu'il n'est pas, l'installation, les thèmes, le branchement à Claude Code. Puis ce qu'on a vu en posant la même demande à Claude, sans puis avec le serveur.

Dépôt, site et documentation relus le 15 novembre 2026. Pour le test, on a branché le serveur MCP et lancé Claude Code. On n'a installé aucun paquet d'Astryx, et le code écrit par Claude n'a pas été lancé. Les commandes viennent du README et de la doc d'Astryx, copiées telles quelles ; on le signale quand une variante est de nous. On n'a aucun lien avec Meta ni avec Anthropic.

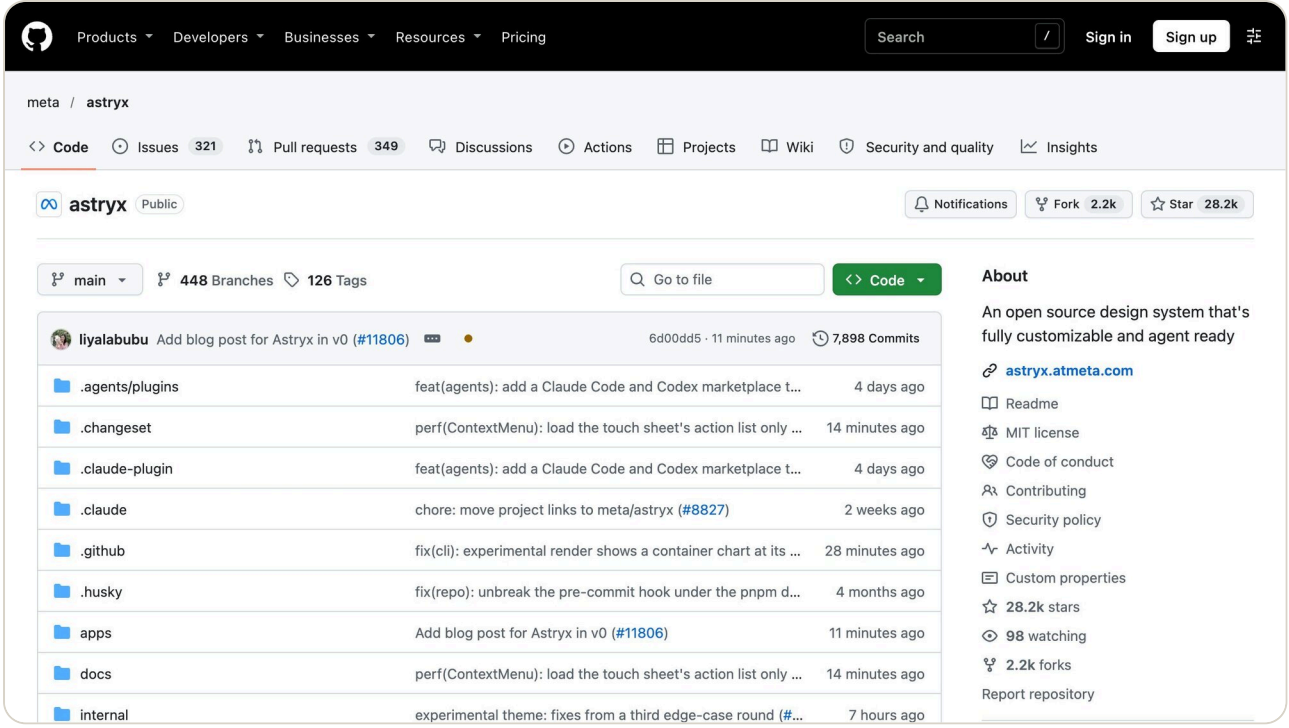
À savoir avant de lire

- Astryx est gratuit, sous licence MIT, usage commercial compris. Claude Code ne l'est pas : il faut un abonnement Claude payant, dès 20 $ HT par mois (Pro), environ 21,70 € TTC.
- Astryx est en bêta (version 0.7.16 du 14 novembre 2026) et ne marche qu'avec React 19 ou plus récent. Les versions Vue, Svelte ou Flutter qu'on trouve sur GitHub ne viennent pas de Meta.
- Les 13 000 applis, c'est le chiffre de Meta pour ses outils internes, et il porte sur le système dont Astryx est la reconstruction. Impossible à vérifier de l'extérieur.
- Le serveur MCP donne la doc à Claude, rien de plus. Il ne lance rien et n'installe rien chez toi.
- Ce que Claude lui demande part chez Meta. Quand il est branché, pas de nom de client ni de donnée sensible dans tes demandes.
- Notre test du 15 novembre : sans le serveur, Claude dit ne pas connaître Astryx et écrit une carte en React pur. Avec, il importe six vrais composants d'Astryx. Le code n'a pas été lancé.

---

1. Ce qu'est Astryx

Le dépôt : github.com/meta/astryx. Licence MIT, 28 182 étoiles le 15 novembre 2026 à 2 h 51, dernier envoi de code à 2 h 50 le même jour. Le site : astryx.atmeta.com.



L'en-tête du dépôt meta/astryx sur GitHub le 15 novembre 2026 vers 3 h : 28,2 k étoiles, 2,2 k forks, licence MIT

Ce qu'il fait

Un design system, c'est la boîte de pièces d'une interface : boutons, cartes, formulaires, tableaux, menus, tous dessinés et codés de la même façon. Astryx en livre plus de 150 sous forme de composants React, avec un thème qui fixe les couleurs, les polices, les arrondis et les espacements. Mode sombre compris. Il ajoute des modèles de pages prêts à remplir et une commande, `astryx`, qui sert la documentation dans le terminal.

Le README dit « 150+ », la page d'accueil du site « Over 180 components ». Dans le dépôt, on compte 185 fiches de composants, sous-composants compris. Les deux chiffres tiennent, selon ce qu'on compte.

Meta l'a pensé pour être lu par une IA autant que par un humain : mêmes noms de props d'un composant à l'autre, une fiche de doc par composant, et deux façons de la donner à un agent, la commande `astryx` et le serveur MCP.

Ce qu'il n'est pas

- Une version finie. Le README l'annonce « Currently in Beta ». Cinq versions sont sorties du 10 au 14 novembre. Certaines pièces ne sont pas encore stables et s'installent depuis la version canary, les graphiques par exemple.
- Une bibliothèque pour Vue, Svelte, Angular ou Flutter. Astryx est construit sur React et StyleX et demande React 19 ou plus récent. Les portages qu'on trouve sur GitHub (astryx-vue, astryx-svelte, astryx_ui pour Flutter) sont des projets d'indépendants, et la plupart écrivent eux-mêmes « Not affiliated ».
- Un générateur d'appli. Il fournit les pièces et leur doc ; c'est Claude, ou toi, qui assemble.
- Le design de Facebook ou d'Instagram. Rien chez Meta ne dit que ces applis tournent dessus.

Ce que veulent dire les 13 000 applis

Meta écrit « 13,000+ apps » dans le README et sur l'accueil du site, « 13,000+ products » dans son billet technique du 29 juin 2026, « How Astryx works ». Ce billet raconte d'où vient le chiffre. Meta tourne sur des milliers d'outils internes, construits par les équipes qui en ont besoin, et chacune refaisait les mêmes pièces : le sélecteur de date, le tableau, le formulaire. Un système commun est né pour ces outils. En huit ans, il est devenu le design system le plus utilisé et le plus gros de Meta.

Astryx est ce système reconstruit sur des briques open source (« that internal system rebuilt on the open-source stack »). Les 13 000 sont donc des outils internes de Meta, qui tournent sur le système d'origine. Le chiffre vient de Meta : personne ne peut le vérifier de l'extérieur.

---

2. L'installer

Il te faut un projet React 19 ou plus récent (Next.js, Vite ou autre). La commande du README installe Astryx, le thème Neutral et StyleX, puis la CLI en dépendance de développement :

À COLLER

```
npm install @astryxdesign/core @astryxdesign/theme-neutral @stylexjs/stylex
npm install -D @astryxdesign/cli
```

Le README donne les mêmes lignes pour pnpm et yarn. Aucun plugin de build à régler : Astryx livre son CSS déjà compilé.

Ensuite, d'après le README du paquet core (le cas de Next.js, le plus simple), trois imports dans ton CSS global, dans cet ordre, `src/app/globals.css` :

À COLLER

```
@import '@astryxdesign/core/reset.css';
@import '@astryxdesign/core/astryx.css';
@import '@astryxdesign/theme-neutral/theme.css';
```

Puis le thème autour de l'appli, `src/app/providers.tsx` :

À COLLER

```
'use client';

import Link from 'next/link';
import {Theme} from '@astryxdesign/core/theme';
import {LinkProvider} from '@astryxdesign/core/Link';
import {neutralTheme} from '@astryxdesign/theme-neutral/built';

export function Providers({children}: {children: React.ReactNode}) {
  return (
    <Theme theme={neutralTheme}>
      <LinkProvider component={Link}>{children}</LinkProvider>
    </Theme>
  );
}
```

Et dans `src/app/layout.tsx`, `<Providers>` autour de `{children}`. Chaque composant s'importe depuis son propre chemin, par exemple `import {Button} from '@astryxdesign/core/Button'`. Les variantes pour Tailwind, Vite et une page sans outil de build sont dans le README du paquet core.

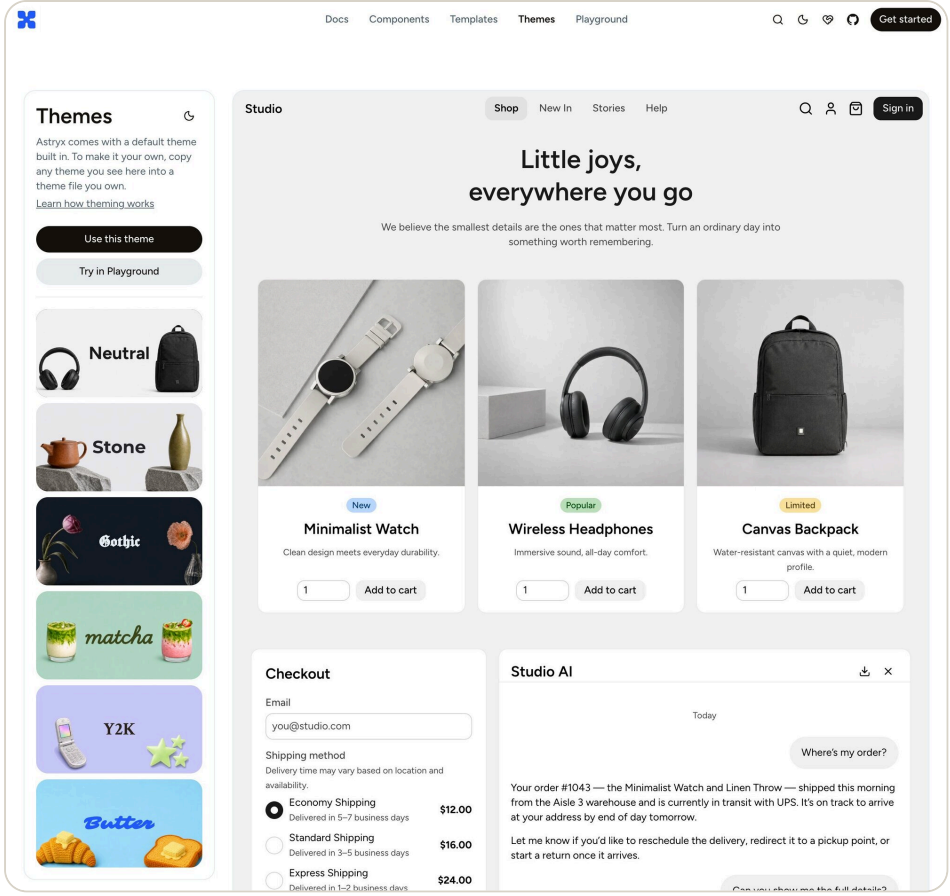
---

3. Les sept thèmes, et comment en changer

Sept paquets de thèmes, tous publiés sur npm en version 0.7.16 le 15 novembre 2026. Les descriptions viennent du README des thèmes :

- Neutral : discret et minimal, police Figtree. Celui des exemples.
- Butter : surfaces jaune beurre, accents bleus.
- Chocolate : chocolat et beige, titres en Fraunces.
- Gothic : sombre seulement, gris bleutés.
- Matcha : palette de verts.
- Stone : tons pierre et ardoise, titres en Montserrat.
- Y2K : pervenche, accents holographiques, esprit années 2000.

Pour les voir sur une vraie page : astryx.atmeta.com/themes. Le 15 novembre, cette page n'en montrait que six : Chocolate manquait au sélecteur, alors qu'il est dans le dépôt et sur npm.



La page des thèmes d'Astryx le 15 novembre 2026 : six vignettes à gauche (Neutral, Stone, Gothic, Matcha, Y2K, Butter), une boutique d'exemple en thème Neutral à droite

Pour changer de thème, le README de Butter dit d'installer son paquet :

À COLLER

```
npm install @astryxdesign/theme-butter
```

Puis d'importer son CSS et de passer son objet au composant Theme :

À COLLER

```
import {butterTheme} from '@astryxdesign/theme-butter/built';
import {Theme} from '@astryxdesign/core/theme';
import '@astryxdesign/theme-butter/theme.css';

function App() {
  return <Theme theme={butterTheme}>{/* Your app content */}</Theme>;
}
```

Même chose pour les autres : remplace butter par le nom du thème. Retire l'ancien thème de ton CSS global ; on n'a pas testé ce qui se passe si deux thèmes restent chargés.

La page Getting Started propose aussi la CLI : `npx @astryxdesign/cli theme list` pour la liste, `npx @astryxdesign/cli theme add neutral` pour en ajouter un. Pour un thème à toi, tu pars d'un thème existant avec `defineTheme({extends: …})` et tu changes couleurs, polices, arrondis ou mouvements. Le guide complet : `npx @astryxdesign/cli docs use-a-theme`.

4. Brancher le serveur MCP à Claude Code

Le serveur tourne chez Meta, à l'adresse https://astryx.atmeta.com/mcp. Rien à installer, ni compte ni clé : notre Claude Code s'y est connecté sans rien demander.

La commande officielle

La doc d'Astryx (page « Working with AI », section « MCP Server ») donne cette commande, à taper dans le terminal, dans le dossier de ton projet :

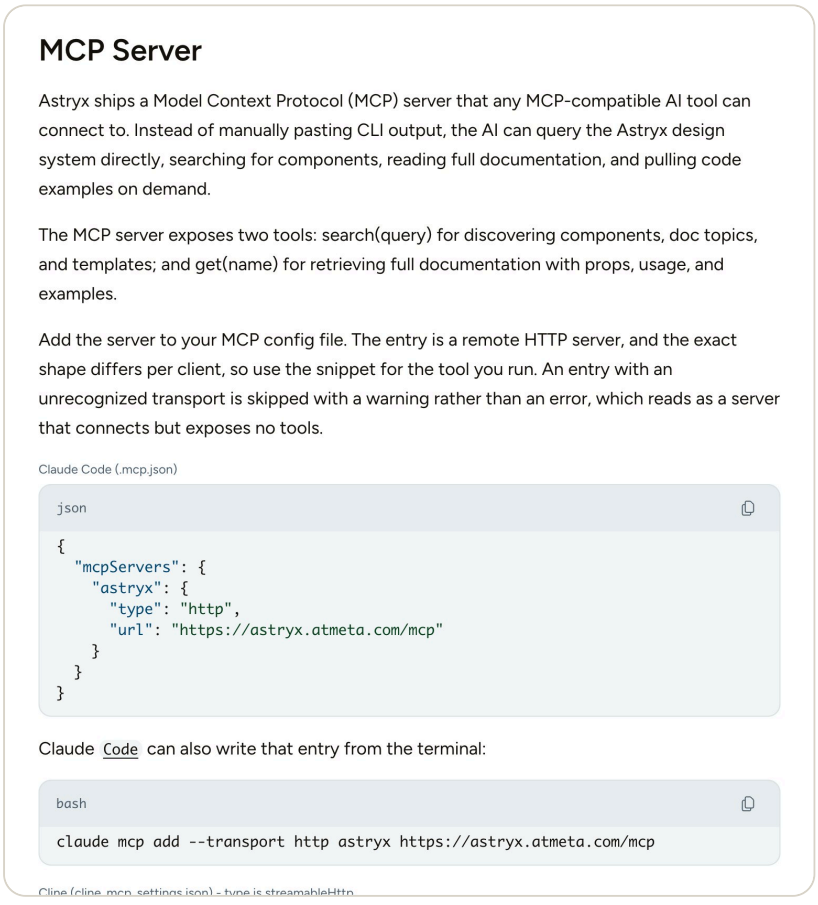
```
À COLLER

claude mcp add --transport http astryx https://astryx.atmeta.com/mcp
```

D'après la doc de Claude Code, sans autre option, elle range le serveur pour ce projet seulement, dans ton fichier `~/.claude.json`. Pour l'avoir dans tous tes projets, notre variante, avec l'option `--scope` de la doc de Claude Code :

```
À COLLER

claude mcp add --transport http astryx --scope user
https://astryx.atmeta.com/mcp
```



MCP Server

Astryx ships a Model Context Protocol (MCP) server that any MCP-compatible AI tool can connect to. Instead of manually pasting CLI output, the AI can query the Astryx design system directly, searching for components, reading full documentation, and pulling code examples on demand.

The MCP server exposes two tools: search(query) for discovering components, doc topics, and templates; and get(name) for retrieving full documentation with props, usage, and examples.

Add the server to your MCP config file. The entry is a remote HTTP server, and the exact shape differs per client, so use the snippet for the tool you run. An entry with an unrecognized transport is skipped with a warning rather than an error, which reads as a server that connects but exposes no tools.

Claude Code (.mcp.json)

json

```
{
  "mcpServers": {
    "astryx": {
      "type": "http",
      "url": "https://astryx.atmeta.com/mcp"
    }
  }
}
```

Claude Code can also write that entry from the terminal:

bash

```
claude mcp add --transport http astryx https://astryx.atmeta.com/mcp
```

Cline (cline_mcp_settings.json) - type is streamableHttp

La section MCP Server de la page Working with AI d'Astryx le 15 novembre 2026 : les deux outils search et get, le bloc .mcp.json pour Claude Code et la commande claude mcp add

Le fichier .mcp.json

L'autre voie de la doc : un fichier `.mcp.json` à la racine du projet. Il se partage avec ton équipe par Git. C'est celui qu'on a utilisé pour notre test, mot pour mot.

À COLLER

```
{
  "mcpServers": {
    "astryx": {
      "type": "http",
      "url": "https://astryx.atmeta.com/mcp"
    }
  }
}
```

La commande officielle avec `--scope project` écrit ce fichier pour toi. Au premier lancement, Claude Code te demande d'approuver un serveur venu d'un `.mcp.json`.

Cursor, Cline, Claude Desktop

- Cursor : le même bloc dans `.cursor/mcp.json`, sans la ligne `"type"`. La doc précise que les serveurs distants n'en prennent pas.
- Cline : le même bloc dans `cline_mcp_settings.json`, avec `"type": "streamableHttp"`.
- Claude Desktop : son fichier de configuration ne prend que les serveurs locaux. Le serveur d'Astryx s'ajoute par Settings, puis Connectors, puis Add custom connector, en collant l'adresse https://astryx.atmeta.com/mcp. Ce sont les intitulés de la doc d'Astryx, en anglais ; on ne l'a pas fait.

La doc prévient : une entrée avec un mauvais type de transport est ignorée avec un simple avertissement. Le serveur a l'air connecté mais n'offre aucun outil.

Vérifier

À COLLER

```
claude mcp list
```

La ligne astryx doit afficher « ✔ Connected ». Dans une session, la commande `/mcp` montre le serveur et ses deux outils.

Les autres voies de Meta

- La CLI et ses consignes pour agents. La page Getting Started fait installer la CLI puis lancer `npx @astryxdesign/cli init`, qui écrit un bloc d'instructions Astryx dans `AGENTS.md` et, pour Claude Code, dans ton `CLAUDE.md`, entre des marqueurs Astryx. Le texte autour n'est pas touché, d'après la doc. On ne l'a pas lancée.
- Une skill pour Claude Code, dans le dépôt depuis le 10 novembre 2026. Elle s'appuie sur la commande `astryx`, à installer avec npm. Pas essayée.

5. Ce que fait le serveur, et ce qu'il ne fait pas

Deux outils, d'après la doc et le code du serveur, publié dans le dépôt :

- search : Claude lui passe une recherche en langage courant (« dropdown menu », « success message »). Il renvoie des résultats courts : nom du composant, description, props principales, ligne d'import, commande d'installation, et si la pièce est stable ou encore en canary. Il cherche aussi dans les sujets de doc et les modèles de pages.
- get : Claude lui donne un nom (Card, Button). Il renvoie la fiche complète : toutes les props avec leur type, les règles d'usage, des exemples de code, ce que le thème peut changer.

Ce qu'il ne fait pas :

- Il ne lance rien sur ta machine, ne lit pas tes fichiers, n'installe aucun paquet.
- Il n'écrit pas le code. Claude lit la doc et décide.
- Il ne contrôle pas le résultat. Si Claude se trompe de prop, tu le découvres en lançant l'appli. Meta le reconnaît dans son billet « Vibe Test » du 27 juillet 2026 : une nuit de tests, ses agents ont inventé des imports et des noms de props sur quelques demandes (« hallucinated imports and prop names on a handful of prompts »).
- Il ne fait pas l'installation à ta place. La commande qu'il renvoie ne cite que `@astryxdesign/core`. Le thème, StyleX, les imports CSS et le composant Theme de la partie 2 restent à faire.

Dans notre test, ses six réponses faisaient chacune de 3 600 à 14 000 caractères, autant de texte en plus à lire pour Claude. La passe avec le serveur a duré 24 secondes contre 20 sans, et Claude Code a estimé son coût à 0,41 $ contre 0,08 $. Avec un abonnement, rien n'est facturé à part : ça puise dans ton quota.

---

6. Notre test, sans puis avec le serveur

On a posé la même demande à Claude Code deux fois, le 15 novembre 2026 à 2 h 40 :

> **À COLLER**
>
> ```
> Construis une carte produit pour une boulangerie : photo, nom du produit, prix,
> et un bouton « Ajouter au panier ». Utilise Astryx, le design system open source
> de Meta. Donne seulement le code React, dans un seul fichier.
> ```

Même configuration pour les deux passes : Claude Code en mode non interactif, sans nos réglages ni nos fichiers d'instructions, recherche web coupée, dans un dossier vide. Seule différence, la seconde passe avait le serveur d'Astryx, branché avec le `.mcp.json` officiel. Même modèle les deux fois : Fable 5.2, celui que Claude Code a pris par défaut dans ce mode.

Sans le serveur

Claude a d'abord fouillé le dossier vide, puis répondu : « Je ne connais pas de design system Meta nommé « Astryx », et il n'y a aucune trace de ce paquet dans le répertoire de travail. Plutôt que d'inventer des imports et des composants, voici la carte en React pur ».

Le code fait 105 lignes, sans aucun import d'Astryx, avec huit couleurs tapées à la main. Le début :

```
À COLLER
const styles = {
  card: {
    width: 320,
    borderRadius: 16,
    overflow: "hidden",
    background: "#fffaf3",
    boxShadow: "0 4px 16px rgba(60, 40, 20, 0.12)",
    fontFamily: "system-ui, -apple-system, 'Segoe UI', sans-serif",
    color: "#3b2a1a",
  },
```

Une carte correcte, qui ne suit aucun thème : pour passer en sombre ou changer de couleur, tout se reprend à la main.

Claude a préféré ne rien inventer. On a fait trois passes sans le serveur cette nuit-là, deux avec Fable 5.2 et une avec Opus 5.5 : aucune n'a inventé de composant Astryx. Deux ont écrit du React pur, la troisième du StyleX, la bibliothèque de styles de Meta.

Avec le serveur

Six appels au serveur avant d'écrire une ligne :

1. search « product card with image, price and button »
2. get Card
3. get Button
4. search « image media aspect ratio »
5. get Text
6. get Stack

Le code fait 77 lignes, importe six composants d'Astryx et ne contient aucune couleur tapée. Les imports :

```
À COLLER
import {AspectRatio} from '@astryxdesign/core/AspectRatio';
import {Button} from '@astryxdesign/core/Button';
import {Card} from '@astryxdesign/core/Card';
import {Stack} from '@astryxdesign/core/Stack';
import {Heading, Text} from '@astryxdesign/core/Text';
```

Le corps de la carte, sous la photo :

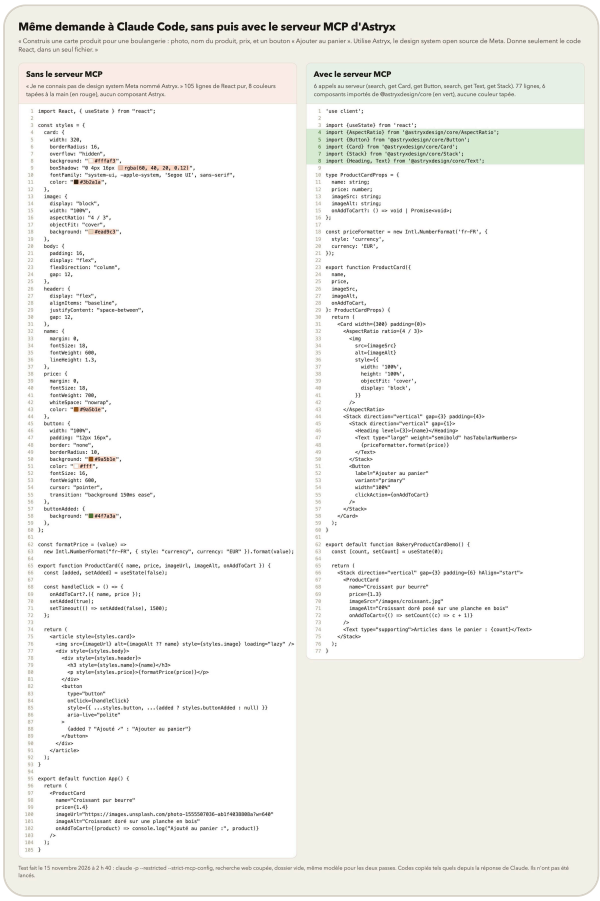
À COLLER

```
<Stack direction="vertical" gap={3} padding={4}>
       <Stack direction="vertical" gap={1}>
         <Heading level={3}>{name}</Heading>
         <Text type="large" weight="semibold" hasTabularNumbers>
           {priceFormatter.format(price)}
         </Text>
       </Stack>
       <Button
         label="Ajouter au panier"
         variant="primary"
         width="100%"
         clickAction={onAddToCart}
       />
     </Stack>
```

Les espacements sont des pas de l'échelle du thème (gap={3}, padding={4}), le texte a un type (large), le bouton une variante (primary). Les valeurs réelles viennent du thème.

On a vérifié chaque composant et chaque prop dans les fiches de doc du dépôt, les fichiers `.doc.mjs` de `packages/core/src/`. Les six composants existent, toutes les props aussi, avec les valeurs utilisées. Heading s'importe bien depuis le chemin de Text.

Ce qui nous a plu : Claude a dit d'où venait chaque choix. Sa réponse précise que les props viennent de la doc d'Astryx, sauf celles de Heading, « reprises des exemples » : il n'avait pas demandé la fiche de Heading.



Nos deux codes du 15 novembre 2026 côte à côte : à gauche sans le serveur MCP, React pur et huit couleurs tapées à la main ; à droite avec le serveur, six composants importés d'Astryx

Ce que le test ne dit pas

- Le code n'a pas été lancé. On ne sait ni à quoi ressemble la carte à l'écran, ni si elle compile.
- L'installation proposée est incomplète. Claude conseille seulement `npm install @astryxdesign/core`. Le README demande aussi le thème, StyleX, les trois imports CSS et le composant Theme (partie 2) : sans eux, les styles d'Astryx ne sont pas chargés.
- Une demande, un modèle. Une passe précédente, dans la même configuration, avait importé VStack au lieu de Stack : la même pièce sous une autre forme. Le résultat varie d'une passe à l'autre.
- La photo pointe vers /images/croissant.jpg : mets la tienne.

---

7. Ce que voit Meta

Le serveur tourne chez Meta. Chaque appel de Claude part vers astryx.atmeta.com avec son contenu. Dans notre test, voici tout ce qui est parti : « product card with image, price and button », « image media aspect ratio », puis les noms Card, Button, Text et Stack. Claude a résumé notre demande en mots-clés anglais, sans le mot boulangerie.

C'est Claude qui rédige ces recherches à partir de ta demande. Si elle contient un nom de client, un produit pas encore annoncé ou des chiffres internes, rien n'empêche qu'un bout parte dans une recherche.

Le site renvoie à la politique de confidentialité de Meta Open Source, en vigueur depuis le 25 avril 2023. Elle prévoit la collecte automatique de l'adresse IP, d'identifiants et des opérations faites sur ses services. Dans le code du serveur publié sur GitHub, les journaux détaillés ne s'activent qu'en développement. Ce que garde l'hébergement, on ne peut pas le voir.

- Pas de données sensibles dans tes demandes quand le serveur est branché.
- Pour un projet client confidentiel, branche-le le temps de la maquette, puis retire-le :

> **À COLLER**
>
> ```
> claude mcp remove astryx
> ```

- Ton code et tes demandes partent de toute façon chez Anthropic, qui fait tourner Claude. Le serveur ajoute Meta à la liste.

---

8. Ce qu'il faut pour t'en servir

- Astryx : gratuit, licence MIT. Tu peux t'en servir pour un client ou un produit payant, le modifier, le redistribuer, en gardant la notice de copyright et la licence. Il est fourni sans garantie.
- Le serveur MCP : gratuit, sans compte.

- Un projet React 19 ou plus récent, avec npm, pnpm ou yarn.
- Claude Code, avec un abonnement Claude payant. L'offre gratuite ne l'inclut pas : la page des tarifs met « Non » en face de Claude Code pour Free. Le 15 novembre 2026, cette page en français affiche Pro à 20 $ par mois sans engagement, 17 $ par mois à l'année (200 $ payés d'avance), hors taxes : « Les prix affichés n'incluent pas les taxes applicables ». Aucun prix en euros n'y figurait ce jour-là. Au taux de la BCE du 13 novembre (1 € = 1,1052 $), 20 $ font environ 18,10 € HT, 21,70 € TTC avec la TVA à 20 % (notre calcul). Le 13 novembre, la même page affichait 18 € HT. Le montant exact en euros s'affiche au paiement.
- Ou Cursor, Cline ou Claude Desktop : la doc d'Astryx donne leur configuration (partie 4). On ne les a pas essayés.

---

9. Une demande type pour démarrer

Dans le dossier d'un projet React 19, serveur branché, colle ceci dans Claude Code en changeant le sujet :

À COLLER

```
Installe Astryx dans ce projet comme l'indique le README de meta/astryx :
@astryxdesign/core, @stylexjs/stylex, le thème neutral, les trois imports CSS et
le composant Theme. Puis construis une page d'accueil pour un fleuriste de
quartier : un bandeau, trois produits en cartes avec photo, nom, prix et bouton,
et un formulaire de contact. Avant d'utiliser un composant, cherche-le avec le
serveur MCP astryx (search puis get) et n'utilise que les props de sa fiche.
Aucune couleur, taille ou marge tapée à la main : seulement les valeurs du
thème. Lance la page et dis-moi ce qui ne compile pas.
```

Pour essayer un autre thème ensuite :

À COLLER

```
Remplace le thème neutral par le thème butter, en suivant le README de
@astryxdesign/theme-butter, puis montre-moi ce qui a changé.
```

Si tu préfères la voie CLI de Meta, voici sa demande de départ, telle quelle, tirée de la page Getting Started. Elle installe la CLI et écrit dans ton CLAUDE.md (partie 4) :

À COLLER

```
Install @astryxdesign/core, @stylexjs/stylex, @astryxdesign/theme-neutral, and
@astryxdesign/cli in this project, then run `npx @astryxdesign/cli init` to set
up agent docs. Read the generated files to learn the conventions.
```

---

10. La méthode

Guide écrit le 15 novembre 2026. Lus par l'API GitHub entre 2 h 51 et 3 h : les chiffres du dépôt meta/astryx, ses dernières versions et derniers commits, sa licence, son README, ceux du paquet core et des thèmes, les fiches de doc des six composants du test et le code du serveur MCP. Relus le même jour : le registre npm, les pages Accueil, Themes, Components, Getting Started et Working with AI du site, quatre billets du blog d'Astryx (l'annonce du 18 juin, « How Astryx works » du 29 juin, « One Month of Astryx » du 22 juillet, « Vibe Test » du 27 juillet), la politique de confidentialité liée par le site, la doc de Claude Code sur MCP, la page des tarifs de Claude et l'article d'aide sur le plan Pro. Le test : Claude Code 2.1.325, le 15 novembre à 2 h 40, avec le journal complet des appels au serveur. Aucun paquet d'Astryx installé, rien cloné, code de Claude jamais lancé. Relu le 23 novembre 2026 avant la mise en ligne : dépôt toujours sous licence MIT (32 400 étoiles), version 0.7.17 du 20 novembre, React 19 toujours demandé par le paquet core et les sept thèmes, commande d'installation du README et commande du serveur MCP inchangées, serveur joint. La page Working with AI propose maintenant aussi une version locale du serveur (`claude mcp add astryx -- npx @astryxdesign/cli mcp`), pas essayée.
