Roboto

LE PLAN · OFFERT

Le Plan Roboto

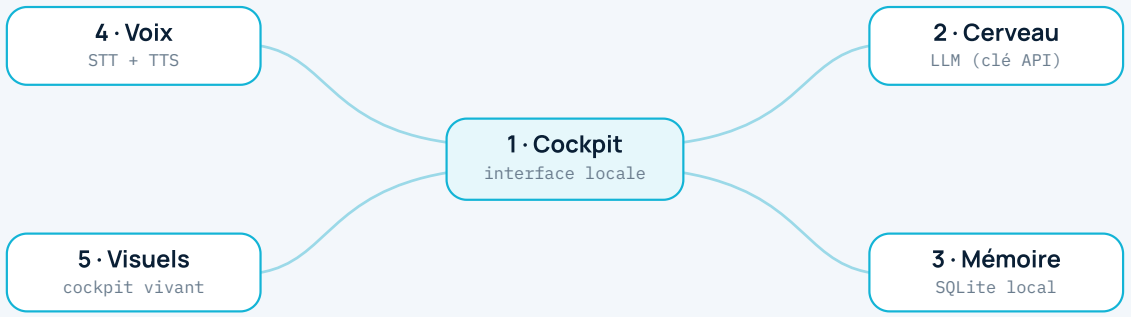
Ton plan pour créer ton propre Roboto : les briques, les outils et les étapes à suivre pour démarrer sans partir dans tous les sens. La vraie carte, pas un teaser.

par Jacky West · jackywest.com

Roboto · Le Plan

Tu as vu mes vidéos de Roboto : un assistant qui m'écoute, se souvient, affiche des choses à l'écran, gère même mes réseaux. La question qui revient le plus : **« comment t'as fait ? »**

Voici la réponse complète, sans bullshit. La vraie carte. À la fin, si tu n'as pas envie de tout monter toi-même, je te montre l'autre option.



Les 5 briques d'un Jarvis local

1 L'interface — le cockpit

→ une page web qui tourne en local

Là où tu parles/écris et où l'assistant affiche ses réponses + des visuels. Le cœur : une connexion temps réel (WebSocket) entre ta page et un petit serveur local.

Piège : vouloir un framework lourd. Une page + un canvas suffisent.

2 Le cerveau — un LLM

→ tu branches une clé (OpenAI ou DeepSeek)

Reste **compatible « API OpenAI »** : tu changes de moteur en une ligne, sans réécrire ton assistant.

Piège : coder en dur le fournisseur → prisonnier d'un prix/quota.

3 La mémoire

`→ stockage local (SQLite) des échanges + des faits`

Tu stockes l'historique ET les **faits importants** (décisions, préférences). Tu réinjectes le contexte pertinent avant chaque réponse. Il ne redemande plus ce qu'il sait déjà.

Piège : tout réinjecter = lent et cher. Sélectionne ce qui compte.

4 La voix — optionnelle

`→ transcription (parole→texte) + synthèse (texte→voix)`

Deux briques : STT et TTS. En cloud, ou en local et `gratuit` avec Piper. Le secret : que ça reste **optionnel**, le texte doit toujours marcher seul.

Piège : rendre la voix obligatoire → micro en panne = assistant mort.

5 Le « vivant » — les visuels

`→ un écran qui s'anime quand il parle`

L'effet « waouh » Jarvis : un visuel qui réagit, des cartes, des graphes. **80 % de l'effet pour 20 % du travail** — et c'est ce qui rend tes vidéos virales.

Piège : la garder pour la fin. Fais-en un peu tôt, ça motive.

> **L'ordre qui marche :** briques 1-2-3 d'abord (interface + LLM + mémoire). Dès que tu dis « bonjour » et qu'il s'en souvient, tu as un assistant. La voix et les visuels s'ajoutent ensuite. Cherche le « ça marche », puis itère.

Le stack exact que j'utilise

BRIQUE	CE QUE J'UTILISE	COÛT
Cockpit	Page web + canvas, serveur local léger, WebSocket	`gratuit`
Cerveau	API compatible OpenAI — OpenAI ou DeepSeek	`ta clé`
Mémoire	SQLite en local (un seul fichier)	`gratuit`
Voix	STT cloud + TTS Piper en local	`gratuit*`
Visuels	Canvas animé, cartes dynamiques	`gratuit`

*Transcription cloud : quelques centimes. Synthèse Piper : 100 % gratuite et locale.

Les 3 pièges qui m'ont coûté des jours

1. Le serveur qui sert du code périmé. En dev, ton serveur peut tourner une vieille version sans que tu le voies. Teste toujours l'effet réel, pas un bout isolé.

2. Router les commandes par mots-clés. « Si le message contient "publie"… » casse au premier cas tordu. Laisse un mini-cerveau LLM décider de l'intention.

3. Tout faire répondre par le gros LLM. Lent et cher. Les petites tâches (accusé, tri) n'en ont pas besoin.

Ta checklist week-end

En un week-end concentré, les briques 1-2-3 sont jouables.

- [ ] **Samedi matin —** une page web locale (champ texte) connectée à un petit serveur (WebSocket).
- [ ] **Samedi aprèm —** brancher une clé LLM (format OpenAI). Premier « bonjour ».
- [ ] **Samedi soir —** stocker les échanges dans SQLite et les réinjecter. Il se souvient.
- [ ] **Dimanche matin —** extraire les « faits » importants en mémoire.
- [ ] **Dimanche aprèm —** un visuel animé qui réagit. L'effet vivant.
- [ ] **Bonus —** la voix (STT + TTS Piper) quand le reste tourne.

LE MULTIPLICATEUR

Construis-le 10× plus vite avec un assistant de code

Tu n'as **pas besoin de savoir coder** pour suivre ce plan. Des outils comme **Claude Code** (Anthropic) ou **Codex** (OpenAI) ouvrent ton projet, comprennent tout le code, et écrivent / corrigent à ta place — tu décris ce que tu veux en français, ils le font. C'est exactement comme ça que j'ai construit Roboto.

- [ ] **Donne-lui ce plan comme carte.** « Construis-moi ça, dans cet ordre — briques 1-2-3 d'abord. » Il a la carte, il déroule.
- [ ] **Avance par petits pas.** Une brique à la fois : « fais la brique 1, puis on teste » > « fais tout ».
- [ ] **Fais-lui TESTER l'effet réel.** « Lance-le et montre-moi le résultat. » Ne valide jamais un « c'est bon » à l'aveugle (piège n°1).
- [ ] **Quand ça plante, colle l'erreur.** « Voilà l'erreur : diagnostique ET corrige. » Il lit, comprend la cause, répare.
- [ ] **Itère.** D'abord « ça marche », ensuite « améliore : plus rapide / plus joli / ajoute la voix ».

> **L'état d'esprit :** tu es le **chef de projet**, l'assistant est ton dev. Tu décides QUOI et tu vérifies que ça marche ; lui s'occupe du COMMENT.

TU AS LE PLAN. VOICI LA MÉTHODE.

Prêt à vraiment le construire ?

Ce Blueprint te donne l'architecture — le **quoi**. Pour le **comment**, j'ai écrit le guide complet : chaque brique expliquée pas à pas, le workflow exact avec Claude Code, et surtout **les 6 prompts prêts à copier-coller** que j'utilise pour construire Roboto. Tu les colles, tu suis le fil, ton agent prend vie.

ROBOTO — LE GUIDE · OFFRE DE LANCEMENT (AU LIEU DE 49 €)

19 €

- ✓ 23 pages denses : l'architecture réelle de Roboto, démontée pièce par pièce
- ✓ 11 chapitres, du concept à la mise en ligne
- ✓ 6 prompts prêts à coller dans Claude Code (du fichier de règles au visage de ton agent)
- ✓ PDF immédiat, accès à vie, futures mises à jour offertes

Le guide qui transforme ce plan en agent qui tourne.

Prendre le guide — 19 €

jackywest.com

Besoin d'être accompagné personnellement sur ton projet digital ? Écris-moi sur jackywest.com/contact.php — je réponds sous 24 h.

Le Plan Roboto — par Jacky West. Garde-le, partage-le.