Quentin Cornu

Développeur Libre

CODER UNE APPLICATION AVEC L'IA

Le guide pour créer une application mobile

Par Développeur Libre, expert en **création d'application iOS** + de 10 millions de vues sur YouTube

DBS

CODER UNE APPLICATION AVEC L'IA

Quentin Cornu

CODER UNE APPLICATION AVEC L'IA

Le guide pour créer une application mobile

DBS

Pour toute information sur notre fonds et les nouveautés
dans votre domaine de spécialisation,consultez notre site web :
www.deboecksuperieur.com

Couverture : Primo & Primo

Création de maquette et mise en page : Ho Thanh Hung

Rue du Bosquet, 7 - B-1348 Louvain-la-Neuve

Dépôt légal :
Bibliothèque nationale, Paris : septembre 2026

Bibliothèque royale de Belgique, Bruxelles : 2026/13647/155 ISBN : 978-2-8073-7668-7

SOMMAIRE

INTRODUCTION

Qui je suis et pourquoi j'ai écrit ce livre

▶ Qui je suis

Janvier 2023, un matin froid d'hiver. J'arrive en salle de classe pour donner cours à mes élèves de Master en digital dans une grande école de la région parisienne. L'objectif de cette semaine va être de leur transmettre toutes les compétences nécessaires pour qu'ils puissent coder leur propre application mobile et la publier sur l'App Store. Je suis confiant, car voilà plusieurs années que je donne ce cours. Je l'ai actualisé à plusieurs reprises, en tenant compte des évolutions majeures du monde d'Apple ces dernières années, mais également des questions de mes étudiants.

Les élèves qui se trouvent en face de moi sont brillants. Ils manient les lignes de code depuis quelques années et ont déjà réalisé des projets semblables à ceux que je vais leur présenter durant les cinq prochains jours. JavaScript, React, Android Studio, ce ne sont pas des mots qui leur sont inconnus. Certains étudiants ont même déjà abordé des problématiques auxquelles je n'avais pas les réponses. Mais, peu importe, car mon objectif est de leur transmettre mon point de vue d'expert en développement iOS, pas d'un autre domaine.

Cette semaine du début du mois de janvier 2023, quelque chose était différent de d'habitude. La disposition de la classe était différente. Ce n'était pas à cause de l'arrivage des tous nouveaux iMacs flambants neufs qui venaient d'être installés dans l'école. L'inhabituel se trouvait dans l'attitude des élèves. Ils me semblent à la fois plus investis, mais moins attentifs. Pourtant, leur participation était anormalement élevée. Chaque question que je posais trouvait sa réponse dans l'une de leurs nombreuses interventions. Leurs questions étaient plus pertinentes que

d'habitude. Peut-être était-ce ça aussi, la joie de l'enseignement : apprécier les disparités dans les compétences de chaque classe.

Comme d'habitude, mes élèves ont accès à leur Mac durant les cours. Car oui, j'ai été traumatisé par les cours d'informatique de deux heures, assis dans un amphithéâtre, à regarder des dizaines de slides défiler, remplies de code en langage C. Peut-être que vous aussi, vous avez connu les examens de code sur une feuille blanche, avec rien d'autre qu'un crayon à papier et une gomme. Je m'en souviens comme si c'était hier… Alors forcément, je suis le premier à encourager mes élèves à pratiquer sur leur machine, à réécrire le code que je leur partage, à personnaliser le projet en même temps que je leur montre comment faire.

Lors de la pause, un élève est venu me voir pour me poser quelques questions. Rien d'anormal jusque-là, puisque j'avais l'habitude de recevoir toutes sortes de profils assoiffés de connaissance. Mais, au lieu de me questionner sur le langage Swift, Xcode ou sur mon aventure entrepreneuriale pour, lui aussi, se lancer en freelance, il m'a posé la question qui changera complètement la direction de ma carrière les années suivantes.

« Monsieur, on a le droit d'utiliser ChatGPT ? »

C'est à ce moment-là que j'ai compris. J'avais vaguement entendu parler de cet outil, sans vraiment comprendre ce qu'il faisait, ou comment on pouvait s'en servir. En effet, voilà plusieurs semaines qu'Open IA avait lancé la version grand public de son modèle GPT 3 : *ChatGPT*. Aujourd'hui, à l'heure où j'écris ces lignes en 2026, tout le monde en parle, tout le monde l'utilise presque quotidiennement et beaucoup d'entre nous ne pourraient plus s'en passer.

Quand j'étais étudiant en école d'ingénieurs, j'ai découvert le monde de la programmation avec le langage C. J'ai alors compris qu'on pouvait créer tout un tas de logiciels et d'applications, dont des applications iOS. Je me suis alors mis en tête d'apprendre le langage Swift pour pouvoir coder ma première application et la mettre sur mon iPhone. Je me voyais déjà montrer à mes amis et mes parents : « Regarde, c'est moi qui l'ai codé ! ». Le lendemain, après les cours, j'ai poussé le bus quelques arrêts plus loin pour atterrir en centre-ville de Caen, proche d'une grande librairie. Le genre de librairie où je me retrouvais souvent à l'approche des concours en prépa, pour acheter ces fameux parpaings à 50 euros pièce. Je me souviens avoir arpenté les rayons sans vraiment trop savoir ce que je cherchais, à la recherche d'un livre qui pourrait m'aider dans ma quête. Informatique, programmation, logiciels… Des étagères entières remplies de livres aux couvertures froides. Soudain, au milieu de cette étale intimidante, je l'ai vu. *Swift pour les nuls*.

Il était là, presque insolent. Sa couverture jaune et son petit logo orange se distinguaient de tous ses collègues. Il était là, comme trônerait la coupe du monde au milieu du stade le soir de la finale. J'ai immédiatement pris le dernier exemplaire restant, filé vers la caisse et repris le prochain bus pour rentrer chez moi. J'avais l'impression de tenir entre les mains une recette secrète qui allait faire de moi le prochain Mark Zuckerberg ou Steve Jobs. À peine rentré, j'ai ouvert mon Mac et commencé à feuilleter les premières pages du livre pour appliquer les concepts, télécharger les logiciels nécessaires et recopier les lignes de code. Malheureusement, mon enthousiasme est vite retombé.

Le langage Swift est né en 2014. Comme toute technologie nouvelle, que ce soit un langage de programmation, un nouveau logiciel ou l'IA générative, les premières années sont toujours chaotiques. Elles bouleversent nos habitudes et révèlent souvent bon nombre de défauts. Pour le Swift, c'est pareil. Les premières versions n'étaient pas si impressionnantes qu'Apple voulait nous faire croire lors de leurs *keynotes*. Il comportait quelques bugs et beaucoup d'améliorations étaient encore possibles. Cela fait partie de la vie d'un langage de programmation. Alors quand j'ai mis ce livre entre mes mains, quelques temps s'étaient déjà écoulés depuis sa rédaction. Et malheureusement, comme si je n'avais pas appris de mes erreurs (apprendre à coder avec un *PowerPoint* et un stylo), je ne m'étais pas rendu compte que le livre était déjà périmé. Plusieurs versions du Swift avaient vu le jour depuis, ce qui rendait le livre obsolète et quasiment inutilisable. La syntaxe avait changé, les mots clés n'étaient plus les mêmes, les boutons sur lesquels je devais cliquer avaient disparu. Le soufflet était retombé.

Après quelques jours à tout remettre en question, je me suis rappelé ce fameux site du zéro. S'ils avaient un cours pour coder un mini jeu en C, pourquoi pas pour coder une application iOS ? Alors j'ai cherché, j'ai retourné le web, en tombant toujours sur des versions plus ou moins récentes, et plus ou moins obsolètes de cours pour apprendre à coder une app. Jusqu'au jour où je tombe sur cette formation : « Coder une application Swift sur iOS 10 ». Génial ! C'était la toute dernière version ! Et en quelques jours seulement, j'avais ma toute première application « Hello World » installée sur mon iPhone.

Cette période marqua le début de mon aventure entrepreneuriale. J'ai alors créé ma chaîne YouTube qui compte, à l'heure où j'écris ces lignes, tout juste 200 000 abonnés. J'y partage toutes mon aventure, de mes premiers pas dans le monde du développement iOS, à mes projets personnels en passant par tous les obstacles qu'un développeur peut rencontrer, dont ceux engendrés par l'intelligence artificielle évidemment.

▶ Pourquoi j'ai écrit ce livre

Durant ma scolarité, *OpenClassrooms* a été d'une aide fondamentale. Ce site m'a aidé à prendre confiance en moi, à comprendre que l'on pouvait quasiment tout apprendre sur internet et surtout à acquérir une certaine forme d'autonomie dans l'apprentissage. Je ne me reposais plus uniquement sur un corps enseignant composé de multiples professeurs qui avaient tous leurs propres façons d'enseigner. J'ai réussi à contourner les limites de l'école traditionnelle.

Aujourd'hui, je vois en l'IA une version d'*OpenClassrooms* sous stéroïdes. Avant, je devais chercher des tutoriels précis, regardes des dizaines de vidéos YouTube en ayant à chaque fois la crainte que celle-ci soit déjà dépassée. Je tombais sur des contenus extrêmement bien expliqués, mais qui n'étaient déjà plus utilisables, car le logiciel ou le langage de programmation avait été actualisé. Avec l'IA, cette phase de recherche et d'incertitude atteint un tout autre niveau ! On peut obtenir une réponse personnalisée, vulgarisée et expliquée comme si on était un enfant de 10 ans. Pour chaque concept flou, on peut obtenir des exemples précis. On peut également obtenir un retour sur notre travail en fournissant un code source pour obtenir une correction ou une piste d'amélioration. On a un partenaire disponible au quotidien, pour nous débloquer dans n'importe quelle situation. L'IA est probablement la plus grande révolution de l'apprentissage du code depuis internet.

Ce livre, je l'ai écrit car je veux vous éviter ce que j'ai moi-même vécu pendant mes études : l'obsolescence permanente. J'ai déjà rencontré ce problème avec le langage Swift et d'autres langages que j'ai voulu apprendre pour me diversifier. Ce livre n'est donc pas une méthode pour apprendre à coder. C'est une méthode, une manière de penser et un système adaptable qui vous permettra d'utiliser l'IA comme un outil pour vous permettre de réaliser toutes les applications que vous avez en tête depuis si longtemps. Je ne voulais pas écrire un livre qui devienne inutilisable dans 2 ans. Je voulais écrire un livre qui vous apprend à évoluer avec les outils. C'est pour cela que vous trouverez peu de références précises de logiciels, de sites web ou d'applications IA. Ce livre est un équilibre subtil entre des

concepts que vous pourrez appliquer à n'importe quel outil IA, et des conseils pratiques pour pouvoir créer quelque chose dès la fin de chaque chapitre.

Le vrai sujet de ce livre ne va donc pas être le code, mais l'autonomie. Ce que je cherchais avant tout, en apprenant avec *OpenClassrooms*, c'est une forme de liberté. Je voulais être libre de me former dans un domaine que j'aimais, pas uniquement les domaines abordés à l'école. Pour moi, le code n'est qu'un véhicule : il me permet de concrétiser mes projets et de passer d'une idée à un produit fonctionnel. Pendant longtemps, créer une app était réservé aux profils techniques. Aujourd'hui, cette frontière a quasiment disparu et le code est devenu accessible à tous.

Que les choses soient claires entre nous. Je ne suis pas un « guru » IA qui vous apprendra les secrets les mieux gardés de ChatGPT ou de ClaudeCode. Je suis l'un de ceux qui a connu la transition entre le « avant » et le « après » IA. J'ai appris à coder à l'ancienne, avec toutes les difficultés de cette méthode. Je sais donc à quel point l'IA change tout lorsqu'il s'agit d'apprendre. Mais je sais aussi qu'il ne faut pas perdre sa parole pour acquis. L'IA n'est pas une encyclopédie. Elle n'a pas la vérité absolue et peut parfois bien cacher son jeu. Je vous transmettrai donc mes meilleurs conseils, acquis au fil des années, pour déceler le vrai du faux. Je vous expliquerai également comment utiliser l'IA correctement pour ne pas tomber dans le piège du tout pour acquis.

Je sais que mes lecteurs auront des profils très variés : des développeurs de longue date, des étudiants qui découvrent le code avec l'IA, ou même des pères de famille qui voient le code comme une science-fiction obscure. À travers ce livre, je veux aussi vous faire comprendre qu'aujourd'hui, il n'est plus nécessaire d'attendre des années avant d'être prêt, pour lancer son projet. Vous n'avez plus besoin d'apprendre à coder pendant plusieurs mois, de suivre une formation d'ingénieur pour créer votre propre application. Aujourd'hui, beaucoup de projets peuvent être créés avec une logique progressive, par expérimentation, petit bloc par petit bloc et entièrement avec l'IA. Et ce, que vous soyez développeur ou non.

Ce livre est un condensé des méthodes, des pièges et des erreurs à éviter quand on développe avec l'IA. J'y ai inclus tous les conseils que j'aurais aimé recevoir à l'époque où j'ai appris à coder seul et surtout à une époque où l'IA telle qu'on la connaît aujourd'hui n'existait pas.

▶ À qui s'adresse ce livre

Je sais que lorsque l'on démarre la lecture d'un nouveau livre, on a toujours cette appréhension de tomber sur un ouvrage ennuyant ou qui ne répond pas à notre besoin. C'est pourquoi je tenais à vous préciser en quelques lignes à qui s'adresse ce livre, pour être sûr qu'il corresponde parfaitement à vos besoins.

Ce livre s'adresse avant tout aux personnes qui veulent créer une application. Peu importe que votre objectif soit de lancer une *startup*, développer un SaaS, publier une application mobile sur l'App Store, créer un petit outil pour votre entreprise, générer un revenu complémentaire ou simplement développer une app utile pour vous et vos proches.

Peut-être que vous rêvez de vivre de vos applications. Peut-être que vous voulez simplement arrêter d'avoir une idée qui traîne dans un coin de votre tête depuis des années sans jamais la concrétiser. Ou peut-être que vous êtes simplement curieux de comprendre ce qu'il est désormais possible de créer avec l'aide de l'IA. Dans tous les cas, vous êtes au bon endroit.

Ce livre a été pensé pour deux profils très différents :
- les débutants qui n'ont jamais réellement développé d'application ;
- les développeurs plus expérimentés qui veulent aller plus vite, structurer leurs projets ou apprendre à collaborer efficacement avec l'IA.

Pendant longtemps, créer une application demandait énormément de connaissances techniques avant même de pouvoir construire quelque chose de fonctionnel. Aujourd'hui, les règles ont changé. Grâce à l'intelligence artificielle, il devient possible de créer des logiciels complets avec beaucoup moins de frictions techniques qu'auparavant.

MISE EN GARDE

Attention, cependant, ce livre n'est pas une promesse magique du style « créer une startup millionnaire en 7 jours sans effort ». L'IA ne remplace pas la réflexion, ni la logique, ni la compréhension du problème utilisateur. En revanche, elle peut considérablement accélérer votre capacité à apprendre, tester, structurer et développer des projets.

Étant moi-même développeur de formation, j'ai subi de plein fouet cette montée en puissance de l'IA. J'ai remis mes compétences techniques en questions. La capacité à coder d'un développeur n'est, en effet, plus très utile aujourd'hui. Tout le monde a accès au code avec un langage naturel. Tout le monde peut écrire le code d'une

application avec plusieurs écrans. Mais ce que l'IA ne peut pas remplacer, c'est la réflexion qui vient autour de la conception de cette application. Le code ne doit plus vous faire peur, mais vous devrez toujours faire ce travail de réflexion, de structuration et de gestion de projet.

C'est justement tout l'objectif de ce livre : vous apprendre à utiliser l'IA comme un accélérateur. Vous allez apprendre à :

- trouver une idée d'application ;
- valider un problème ;
- définir un MVP ;
- structurer vos fonctionnalités ;
- créer vos premiers *mock-up* ;
- choisir les bons outils ;
- développer votre application à l'aide de l'IA ;
- publier votre projet ;
- trouver vos premiers utilisateurs ;
- et surtout construire quelque chose de concret.

Le plus important, c'est que vous n'avez plus besoin d'être un expert pour commencer. Lorsque certains concepts techniques seront nécessaires, nous prendrons toujours le temps de les expliquer simplement. Pour les développeurs les plus avancés, les différents exemples et méthodes présentés dans ce livre pourront facilement être adaptés à des projets beaucoup plus ambitieux. Car au fond, peu importe votre niveau actuel : ce qui compte vraiment, c'est votre capacité à construire et publier quelque chose de réel.

▶ Le guide pratique pour créer une app

Ce livre n'a pas été conçu comme un livre « théorique » que l'on lit passivement dans son canapé avant de passer à autre chose. Je l'ai conçu comme un guide pratique. L'objectif n'est pas simplement que vous compreniez les concepts. L'objectif, c'est que vous construisiez réellement votre application en même temps que vous avancez dans les chapitres. Autrement dit : **ce livre fonctionne beaucoup mieux si vous pratiquez immédiatement.**

Lorsque nous parlerons d'idée, essayez immédiatement de définir la vôtre. Lorsque nous construirons les *User Flows* (on verra ce que c'est, pas de panique), faites-les pour votre projet. Lorsque nous créerons des *mockups*, ouvrez une feuille blanche ou *Figma* et dessinez vos écrans. Lorsque nous développerons l'application, codez en même temps. Plus vous appliquerez les concepts au fur et à mesure, plus ce livre deviendra puissant.

J'ai volontairement structuré les chapitres dans l'ordre logique de création d'un produit :

1. comprendre l'IA et apprendre à collaborer avec elle ;
2. trouver et valider une idée ;
3. construire un MVP ;
4. concevoir les écrans ;
5. choisir les bons outils ;
6. développer l'application ;
7. publier le projet ;
8. trouver ses premiers utilisateurs.

L'idée est simple : si vous suivez le livre étape par étape, vous allez progressivement transformer une idée floue en une vraie application fonctionnelle. À la fin de ce livre, votre application sera publiée sur l'*App Store*, le *Play Store* ou accessible sur le web.

Tout au long du livre, vous trouverez également des exemples concrets, des schémas, des méthodes, des conseils pratiques, des erreurs à éviter, des exercices de mise en pratique et surtout énormément d'exemples d'utilisation de l'IA. Car ce livre ne parle pas uniquement de développement. Il parle surtout de collaboration avec l'IA. Vous apprendrez notamment :

- comment structurer vos prompts ;
- comment utiliser l'IA pour générer du code ;
- comment vérifier ses réponses ;
- comment accélérer votre apprentissage ;
- comment créer des *wireframes* ;
- comment débogguer ;
- comment rédiger de la documentation ;
- ou encore comment organiser votre projet.

Mais gardez toujours une chose en tête : **l'IA est un assistant, pas un pilote automatique.** Le but n'est pas de déléguer entièrement votre cerveau à une machine. Le but est d'utiliser l'IA pour aller plus vite tout en restant capable de comprendre ce que vous construisez. C'est pour cette raison que ce livre mélange en permanence : réflexion produit, logique utilisateur, méthodes de développement et collaboration avec l'IA.

Enfin, n'essayez pas de tout rendre parfait dès le début. Beaucoup de personnes abandonnent leurs projets parce qu'elles veulent immédiatement créer une application parfaite. Ce livre suit une philosophie différente : avancer vite, apprendre en construisant et publier le plus tôt possible. Parce qu'aujourd'hui, avec l'IA, le

plus difficile n'est plus forcément de coder une application. Le plus difficile, c'est d'aller jusqu'au bout.

L'IA : Comprendre, vérifier et accélérer

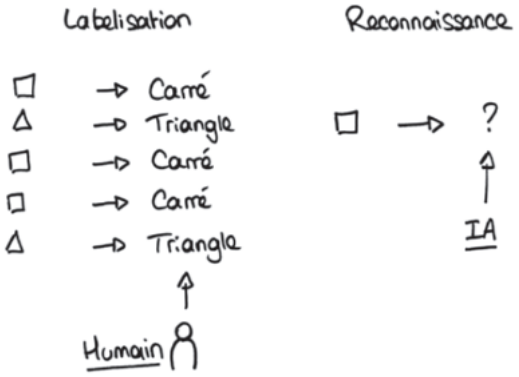
▶ Comment fonctionne un LLM

La nouvelle ère de l'IA

L'Intelligence Artificielle n'est pas une nouvelle notion. En 1950, Alan Turing pose la question fondatrice : *« Can machines think? »*. Pour répondre à cette question et savoir si les machines peuvent réellement penser, il propose un test simple. Un humain discute avec deux interlocuteurs cachés derrière un écran. L'un des deux interlocutrices est un humain, l'autre est une machine. Si la personne qui discute avec ses deux interlocuteurs n'arrive pas à discerner l'humain de la machine, alors on considère que oui : la machine peut penser. Depuis que le test de Turing a été inventé, plusieurs machines l'ont subi à de nombreuses reprises, avec un taux de réussite plus ou moins élevé. Même si les résultats s'améliorent d'année en année, à mesure que les ordinateurs se complexifient, on ne pouvait pas jusqu'à ce jour considérer qu'une machine pouvait réellement penser. Mais l'Intelligence Artificielle est entrée dans une nouvelle dimension depuis quelques années.

Depuis les années 2000 et jusqu'aux années 2010, l'Intelligence Artificielle était relativement limitée à des usages simples et presque déterministes. Elle tirait son « intelligence » d'une méthode d'apprentissage que l'on appelle **l'apprentissage supervisé**. Imaginez cet apprentissage comme un enfant de maternelle qui apprend à lire. Un humain explique à cet enfant comment reconnaître les lettres : il lui montre un gribouillis en lui indiquant que c'est un « A », puis un autre gribouillis : le « B » et ainsi de suite. À force de rencontrer plusieurs dessins qui ressemblent plus ou moins à la même chose, l'enfant finira par comprendre à quoi ressemble chaque lettre de l'alphabet. Cette même méthode sera appliquée durant les premières années de sa vie, pour qu'il apprenne à reconnaitre un chat, un chien, un lion, un crocodile ou encore un avion et une montgolfière. On dit de cet apprentissage qu'il est « supervisé » car un adulte (humain) doit annoter un ensemble de données pour que le programme (ou l'enfant) puisse associer un mot à une image. C'est exactement le principe du *Machine Learning*. Le *Machine Learning* est une forme d'intelligence artificielle dans laquelle on injecte une quantité importante de données étiquetées. Ensuite, grâce à un une série de calculs complexes — que nous n'allons pas détailler dans cet ouvrage, je vous rassure

— le programme peut ainsi créer sa propre intelligence et reconnaître une nouvelle image, un nouveau mot, un nouveau son et l'interpréter comme un concept, un mot ou une phrase.



D'ailleurs, je suis certain que vous avez déjà dû remplir un *CAPTCHA* pour prouver que vous êtes bien un humain. Les tests *CAPTCHAs* font justement partie de la famille des tests de *Turing*. En plus, j'ai une mauvaise nouvelle pour vous : on vous a menti depuis le début. Les petites images sur lesquelles vous cliquiez pour sélectionner un bus ou un vélo n'étaient en fait pas pour vérifier votre identité humaine. Au contraire, vous avez nourri l'intelligence artificielle de Google en étant vous-même cet instituteur qui indique « ceci est un vélo » ou « ceci est un bus ». Google était votre élève de maternelle. Et puis, à votre avis, pourquoi deviez-vous toujours repérer les piétons, les bus ou les vélos ? Pourquoi on ne vous demande jamais de reconnaître une banane ou une pomme ? Tout simple pour nourrir l'une des IA les plus complexes à développer : celle qui va bientôt (si ce n'est pas déjà fait) donner vie à des millions de voitures autonomes.

Cependant, l'apprentissage supervisé limite considérablement les usages de l'intelligence artificielle à des cas d'utilisation très restreints. On l'utilise principalement pour de la reconnaissance vocale ou de la traduction par exemple. Cette forme de technologie ne doit son intelligence qu'à l'étiquetage des données et ne réfléchis pas réellement. Une IA nourrie à l'apprentissage supervisé peut être très douée dans toutes les situations qu'elle a déjà rencontrées. Mais elle devient très mauvaise, voire dangereuse, dans des situations nouvelles pour elle, ou celles qu'un humain pourrait comprendre de manière intuitive. Par exemple, on pourrait entraîner un robot humanoïde à faire un café et à nous l'apporter dans le lit au petit matin. Pour cela, il faudrait lui expliquer à plusieurs reprises comment faire le café, quels placards ouvrir, comment moudre les grains, sur quel bouton

appuyer, comment attraper la tasse et dans quelle direction se tourner pour monter l'escalier et trouver la porte de notre chambre. Cela ne se ferait pas du premier coup. Le robot ferait des erreurs. Il se tromperait de placard, il appuierait sur le mauvais bouton et il ferait surement tomber quelques tasses dans l'escalier. Mais, au bout de plusieurs essais, durant lesquels on aura indiqué au robot les erreurs qu'il aura faites (apprentissage supervisé), il arrivera à vous rapporter le café tous les matins avec un taux de réussite de quasiment 100 %.

Sauf qu'un beau jour, vous décidez de refaire toute votre cuisine pour lui donner un style plus moderne. Comme il vous reste un peu de budget, vous craquez pour cette toute nouvelle machine à café révolutionnaire qui faire le café toute seule — enfin, il faut quand même ajouter les grains et appuyez sur l'écran tactile pour choisir votre boisson préférée. Et là, c'est la panique ! Votre meilleur ami humanoïde est complètement perdu : il n'arrive plus à trouver où est rangé le café, il ne comprend même plus comment allumer la machine et vous devez tout lui réapprendre. **Voici la plus grande faiblesse de l'apprentissage supervisé : il est doué, jusqu'à ce qu'il rencontre quelque chose de nouveau.**

Mais heureusement, depuis le début des années 2020, on assiste à l'émergence d'une nouvelle forme d'IA qui va probablement bientôt vous permettre de changer de cuisine toutes les semaines, tout en continuant de recevoir votre café au bord du lit. Vous avez sûrement déjà entendu parler des LLM ? *Large Language Models*. Les nouvelles Intelligences Artificielles comme *GPT*, *Opus*, *Sonnet* ou encore *Gemini* sont des **IA génératives**. Elles utilisent un modèle qui leur permet de générer du texte en se basant sur des statistiques (ça fait moins rêver, n'est-ce pas ?). Quand on lit un texte, nous, humains, savons que le dernier mot n'est pas forcément le plus important de la phrase. Mais les machines peinent énormément avec ce concept. Pour résoudre ce problème, on utilise ce qu'on appelle : le mécanisme de l'attention. Le principe, c'est d'associer un poids à chaque mot d'une phrase ou d'un bloc de texte. Ainsi, les IA prédictives peuvent comprendre le contexte global d'un texte et générer une suite de mots beaucoup plus logique et cohérente, tout en tenant compte du contexte initial. C'est une explication très schématisée, mais réaliste du fonctionnement d'un LLM *(Large Language Model)*. Alors, comment calculer le poids de chaque mot et comment déterminer la probabilité d'apparition du prochain mot dans une phrase ?

Pour calculer la probabilité qu'un mot apparaisse après un autre pour compléter une phrase, les algorithmes de LLM se nourrissent de quantités astronomiques de données rédigées par l'humain. Elle analyse des sites, des livres, ou tout un tas d'autres textes pour comprendre quels mots paraissent derrière quels autres mots.

C'est ce qu'on appelle la phase de « *pre-training* » ou de pré-apprentissage. Donc la plus grande différence avec les modèles d'entraînement dont nous avons parlé précédemment (l'apprentissage supervisé) c'est qu'**un LLM utilise un apprentissage non supervisé** : aucune intervention humaine n'est nécessaire pour annoter chaque donnée fournie à l'IA. Elle peut donc s'entraîner toute seule, ne nécessite plus de temps humain et peut donc apprendre beaucoup plus rapidement. Elle va reconnaitre certains schémas répétitifs et comprendre les formulations courantes, dans le but d'imiter au mieux le style d'écriture qu'un humain aurait très bien pu utiliser. Il est donc important de comprendre qu'**un LLM ne réfléchit pas, il ne comprend pas réellement le sens d'une phrase et ne peut pas créer un raisonnement logique**. Il se contente simplement de compléter une phrase de manière la plus probable possible. Nous reviendrons par la suite sur cette notion très importante à comprendre pour pouvoir utiliser l'IA dans le développement de nos applications.

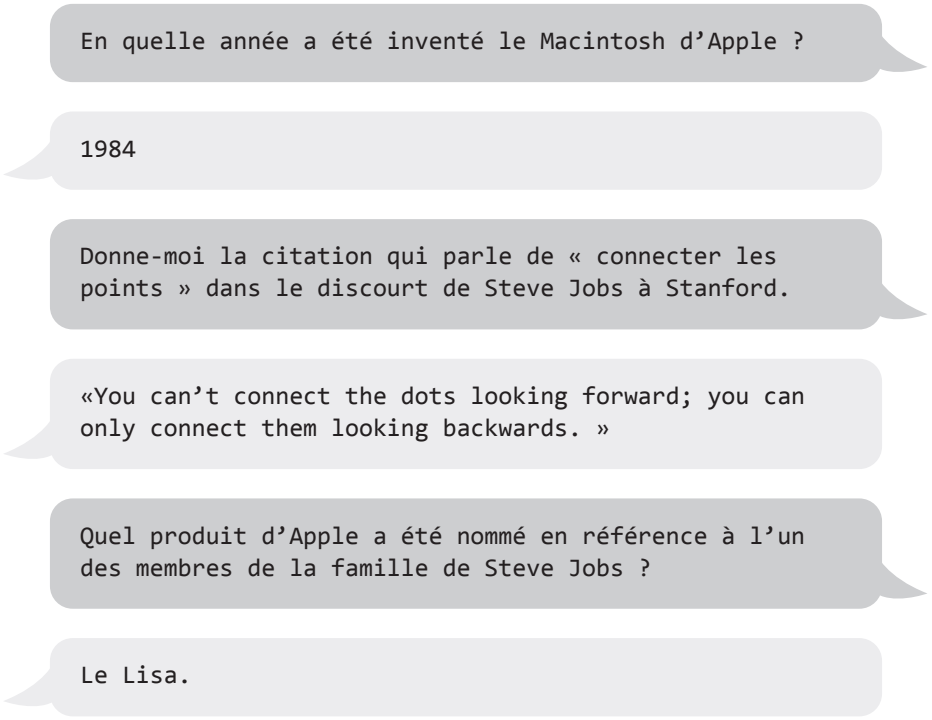
L'IA n'est pas un logiciel traditionnel

La conséquence principale du fonctionnement probabiliste d'un LLM, c'est qu'il n'est pas un logiciel comme les autres. Les logiciels, comme on les connaissait jusqu'à aujourd'hui, sont des logiciels déterministes et répétables. S'ils sont bien développés et qu'ils ne comportent pas trop de bugs, ils fournissent une même sortie pour une même entrée. Si on clique sur un bouton sur *Photoshop*, il génère la même action. Si on lance la même commande dans le Terminal, on obtient le même résultat. De plus, les logiciels classiques sont souvent fournis avec un mode d'emploi ou une documentation. N'importe quel logiciel ou langage de programmation est accompagné d'une documentation pour fournir des explications déterministes sur chaque fonction qu'il contient. Avec un LLM, vous pourriez poser cent fois la même question, il est très peu probable, voire impossible, que vous obteniez deux fois la même réponse !

Comprendre pourquoi l'IA hallucine

L'une des plus grandes faiblesse de l'IA générative (le fait qu'elle ne soit pas déterministe) est aussi sa plus grande force. Comme nous venons de le voir, elle est très puissante pour imaginer la suite d'une phrase, tout en donnant l'illusion que ce texte a été rédigé par un humain. Ce qui est tout à fait logique, puisqu'elle a appris à « parler » en analysant des textes, eux-mêmes rédigés par des humains et que son but est de les imiter. Elle est donc très puissante pour compléter un texte et répondre à une question, en se basant que des schémas statistiques qu'elle a retrouvés dans ses données d'entraînements. Mais son seul et unique but, c'est

de compléter ces phrases. Elle va donc tout faire pour, quitte à générer des informations biaisées, voire fausses. L'IA peut générer une réponse fausse, en donnant l'impression qu'elle est vraie, tant l'IA a « confiance » en elle. Car en effet, elle n'a aucune idée de ce qui est vrai ou faux, elle se contente juste de calculer des probabilités. Et c'est là son plus gros défaut : **l'IA générative peut halluciner**.



Un LLM optimise la vraisemblance de la suite des mots, pas la vérité

L'IA n'est pas programmée pour nous répondre «Je ne sais pas ». En donnant l'illusion qu'elle a de la mémoire, elle peut vite nous perdre dans ses réponses et nous contraindre à les prendre pour acquises. Alors comment utiliser l'IA à notre avantage ? Comment faire de cette faiblesse une force et l'utiliser pour développer des applications complètes, sans qu'elle ne nous mène en erreur et nous fasse perdre plus de temps qu'elle ne nous en fait gagner ? En réalité, on se rend compte que bon nombre de tâches de notre travail sont des tâches créatives. Ce sont des tâches qui ne nécessitent pas de réponses vraies ou fausses, mais plutôt un travail nuancé qui ne constitue pas une vérité absolue. Prenons l'exemple du code : existe-t-il vraiment une réponse unique à chaque problème ? Doit-on écrire un code bien précis pour obtenir une page web ou un écran d'application ?

Non, évidemment, même pour un problème très simple, il existe des dizaines de combinaisons de code possibles pour obtenir le même résultat. Ces combinaisons augmentent exponentiellement à mesure que le projet devient complexe. Coder, c'est donc mélanger sa créativité à un ensemble de règles et de schémas, pour respecter la syntaxe du langage. Et justement, **l'IA est un excellent outil lorsqu'il s'agit de créativité**.

Utiliser un modèle de LLM comme *GPT*, *Opus* ou *Gemini* est un excellent moyen de synthétiser, compresser ou organiser des données. L'IA peut vous permettre de résumer une documentation pour vous expliquer en quelques mots comment utiliser un *framework* dans votre projet. Elle peut organiser vos différentes fonctionnalités en grandes catégories, pour vous aider à organiser votre app. Elle peut compresser plusieurs centaines de lignes de code pour vous expliquer, en langage humain (pas en code) ce qu'il fait réellement. L'IA sera donc très utile pour lire, générer de la documentation et synthétiser des informations complexes. Nous verrons par la suite comment tirer profit de cette force que nous procure l'IA pour l'utiliser concrètement dans la création de nos applications.

▶ L'humain dans la boucle

Maintenant que vous comprenez un peu mieux comment fonctionne un LLM, vous devriez percevoir à la fois son immense puissance… et ses limites. C'est probablement le point le plus important de ce livre. L'IA peut vous aider à apprendre, à réfléchir, à structurer, à coder, à corriger, à améliorer. **Mais elle ne doit jamais devenir une source de vérité absolue**. Elle doit être considérée comme un tuteur, un coach, un professeur particulier disponible à tout moment, mais pas comme un oracle qui aurait toujours raison.

C'est une nuance essentielle. Un professeur peut vous expliquer un concept, vous donner un exercice, corriger votre travail et vous aider à progresser. Mais vous ne devez pas apprendre à devenir dépendant de lui. L'objectif n'est pas de lui demander les réponses à chaque problème, mais d'utiliser son accompagnement pour développer vos propres compétences. Avec l'IA, c'est exactement la même chose. Si vous lui déléguez tout sans chercher à comprendre, vous irez peut-être vite au début, mais vous deviendrez incapable de vérifier, de corriger ou d'améliorer ce qu'elle produit. C'est là qu'intervient l'idée centrale de ce chapitre : **vous devez rester l'humain dans la boucle**.

Cela signifie que vous pouvez utiliser l'IA à chaque étape de votre travail, mais que vous restez responsable de la direction, de la validation et des décisions finales. L'IA peut proposer, générer, organiser, expliquer, reformuler. Mais c'est vous qui

devez vérifier, choisir et comprendre. Autrement dit, vous ne devez pas disparaître du processus. Vous devez apprendre à **collaborer avec elle**. Et pour cela, je vous propose un cadre de travail en quatre principes que j'ai découverts dans le livre *Co-Intelligence* d'Ethan Mollick.

Principe 1 : Inviter l'IA à la table

Le premier principe est simple : invitez l'IA dans votre travail aussi souvent que possible. Pas forcément pour qu'elle fasse tout à votre place, mais pour apprendre à comprendre ce qu'elle peut réellement vous apporter. Trop de personnes utilisent l'IA uniquement pour une ou deux tâches évidentes : écrire un email, résumer un texte, corriger une faute. C'est déjà utile, mais c'est loin d'être suffisant. Pour progresser, vous devez expérimenter. Vous devez tester l'IA dans différentes situations : planifier une fonctionnalité, expliquer une erreur, relire du code, proposer une architecture, analyser une maquette, améliorer une *landing page*, générer un test utilisateur.

Personnellement, j'utilise l'IA absolument tous les jours, que ce soit pour m'aider dans le développement d'un app, pour gérer mon potager ou pour choisir une nouvelle paire de chaussures de sport. Chaque expérimentation vous apprend quelque chose. Vous découvrez ce que l'IA fait très bien, ce qu'elle fait moyennement, ce qu'elle invente parfois, et les situations dans lesquelles elle a besoin d'un cadre beaucoup plus précis. C'est en l'utilisant régulièrement que vous allez construire votre propre intuition.

Principe 2 : Être l'humain dans la boucle

Le deuxième principe est sans doute le plus important : vous devez rester l'humain dans la boucle. Pourquoi ? Parce que l'IA n'est pas conçue comme un logiciel traditionnel qui applique toujours les mêmes règles et produit toujours le même résultat. Comme nous l'avons vu précédemment, elle génère une réponse **probable**. Elle cherche à produire une suite cohérente de mots. Et, très souvent, cette réponse donne l'impression d'être vraie. Le problème, c'est qu'elle peut se tromper avec énormément d'assurance. Elle peut inventer une information, justifier une mauvaise réponse, proposer une solution qui semble élégante mais qui ne fonctionne pas réellement. Et plus vous insistez dans une mauvaise direction, plus elle risque de vous suivre, simplement parce qu'elle cherche aussi à vous satisfaire.

C'est pour cela que vous devez mettre en place un système de vérification. L'objectif n'est pas d'avoir peur de l'IA, mais d'être serein parce que vous savez quoi contrôler. Quand elle vous donne du code, vous vérifiez qu'il compile, qu'il

respecte votre architecture, qu'il ne modifie pas des fichiers hors périmètre. Quand elle vous donne une information factuelle, vous vérifiez sa source. Quand elle vous propose une décision produit, vous la confrontez à vos objectifs, à vos utilisateurs et à vos contraintes. **Être l'humain dans la boucle, ce n'est donc pas tout faire soi-même. C'est garder le rôle de pilote.** Vous laissez l'IA accélérer l'exécution, mais vous gardez la responsabilité de la trajectoire.

Principe 3 : Traiter l'IA comme une personne… sans oublier que ce n'en est pas une

Le troisième principe peut sembler étrange : pour bien travailler avec l'IA, il est souvent utile de la traiter comme une personne. Mais attention, il faut immédiatement ajouter une nuance : l'IA n'est pas une personne. **Elle ne pense pas, ne ressent rien, ne comprend pas le monde comme nous le comprenons**. Quand on dit qu'elle « réfléchit », qu'elle « comprend » ou qu'elle « décide », on utilise une métaphore pratique, pas une vérité scientifique.

Pourtant, dans l'usage quotidien, cette métaphore est utile. L'IA fonctionne mieux quand vous lui donnez un rôle clair. Si vous lui demandez simplement « aide-moi à améliorer ce texte », vous obtiendrez souvent une réponse générique. Si vous lui dites : « Agis comme un éditeur pédagogique spécialisé dans les livres techniques pour débutants, et aide-moi à rendre ce passage plus clair sans changer mon ton », vous obtiendrez une réponse beaucoup plus pertinente.

Ce n'est pas magique. Dire à l'IA « sois Steve Jobs » ne va pas vous transformer en génie du produit. En revanche, lui donner une posture, un contexte et des contraintes permet de guider sa réponse. Vous lui donnez un angle. Vous lui expliquez quelle casquette elle doit porter pour vous aider.

Principe 4 : Partir du principe que c'est la pire IA que vous utiliserez

Le quatrième principe est le plus rassurant : partez du principe que l'IA que vous utilisez aujourd'hui est la pire IA que vous utiliserez dans votre vie. Cette idée change complètement votre rapport aux outils. Oui, les modèles actuels ont des défauts. Oui, ils hallucinent. Oui, ils oublient parfois le contexte. Oui, ils peuvent produire du code imparfait. Mais ils ne feront que s'améliorer.

Cela signifie que les méthodes que vous allez apprendre dans ce livre ne seront pas rendues inutiles par les prochaines générations d'IA. Au contraire, elles deviendront encore plus puissantes. Si vous apprenez aujourd'hui à bien structurer vos demandes, à vérifier les réponses, à découper un projet, à garder un humain dans la boucle, alors chaque amélioration des modèles rendra votre méthode plus efficace.

C'est pour cela que ce livre ne cherche pas à vous apprendre uniquement sur «quel bouton cliquer » dans tel outil à un instant précis. Les interfaces vont changer. Les modèles vont évoluer. Les noms des outils sont différents. Mais le cadre de collaboration restera valable : définir une intention claire, fournir du contexte, poser des contraintes, vérifier les résultats, apprendre à itérer.

En réalité, votre objectif n'est pas de devenir expert d'un outil. Votre objectif est de devenir compétent dans votre manière de collaborer avec l'IA. Et c'est cette compétence qui fera la différence.

Le bon état d'esprit

Si je devais résumer ce cadre en une seule phrase, je dirais ceci : **utilisez l'IA comme un accélérateur, pas comme une béquille**. Une béquille vous permet d'avancer sans utiliser pleinement vos propres capacités. Un accélérateur, lui, amplifie une direction que vous avez déjà choisie. C'est exactement ainsi que vous devez voir l'IA. Elle ne remplace pas votre jugement : elle l'augmente. Elle ne remplace pas votre apprentissage : elle le rend plus rapide. Elle ne remplace pas votre responsabilité : elle vous oblige même à l'assumer davantage.

Tout au long de ce livre, nous allons donc utiliser l'IA intensivement. Nous allons l'inviter à chaque étape sans jamais lui donner le contrôle total. Vous allez apprendre à lui demander de l'aide, à la challenger, à lui faire expliquer ses choix, à vérifier ses réponses, à corriger ses erreurs et à garder une vision claire de votre projet. C'est ce cadre qui vous permettra de coder avec l'IA sans vous faire dépasser par elle.

Partie 1

SE FIXER UN OBJECTIF CLAIR

Vous avez une idée d'application, mais vous ne savez pas par où commencer ?

Pendant longtemps, créer une application semblait réservé aux développeurs expérimentés. Il fallait apprendre un langage, comprendre des concepts techniques, ... Et surtout ne pas abandonner en chemin !

Aujourd'hui, l'intelligence artificielle change complètement la donne. Elle peut vous aider à clarifier votre idée, structurer votre projet et même écrire du code pour vous ! Mais pour obtenir un vrai résultat, il ne suffit pas de demander à l'IA de "coder une app", il faut savoir lui parler, guider ses réponses, comprendre ce qu'elle produit et construire avec méthode.

Dans ce livre, **vous allez découvrir comment passer d'une simple idée à une application concrète**, en utilisant l'IA comme copilote. Vous apprendrez notamment à :

- transformer une idée floue en projet clair ;
- concevoir les écrans et les fonctionnalités essentielles de votre application ;
- dialoguer efficacement avec l'IA pour générer du code propre ;
- comprendre les bases du développement sans vous perdre dans la technique ;
- corriger les erreurs, améliorer votre app et avancer sans rester bloqué.

Ce livre vous propose une méthode moderne, progressive et accessible pour créer votre première application avec l'IA, que vous soyez débutant ou développeur expérimenté.

Quentin Cornu, développeur iOS, formateur et créateur de contenu suivi par plus de 200 000 personnes sur YouTube, vulgarise le code depuis plusieurs années pour le rendre plus simple, plus concret et plus motivant.

24,90 €
ISBN : 978-2-8073-7668-7
9 782807 376687
livre écoresponsable