PASSER DE L'IDÉE AUX STORES : COMMENT BIEN LANCER SON APPLICATION MOBILE EN 2025 ?

FLORIAN FAGNIEZ
JULIAN CLEMOT

EBOOK - NOVEMBRE 2024

ippon

SOMMAIRE

FLORIAN FAGNIEZ

J'occupe depuis 3 ans le poste de Développeur Mobile chez Ippon Technologies à Nantes. Mon rôle consiste à transformer vos idées en solutions concrètes et accessibles pour smartphones et tablettes.

JULIAN CLÉMOT

Je suis Lead Mobile chez Ippon Technologies depuis bientôt 3 ans. Au quotidien, j'accompagne les entreprises nécessitant mes compétences en développement Mobile, que ce soit à travers la matérialisation d'un besoin en prototype ou de l'industrialisation d'une application grand public.

Nous sommes Mentors du programme BlackBelt, un programme dédié au développement des compétences en interne. Nous accompagnons celles et ceux qui souhaitent améliorer leur expertise mobile et approfondir leur maîtrise.

POURQUOI FAIRE **DU MOBILE ?**

> L'ESSOR DU DÉVELOPPEMENT MOBILE S'EST AMORCÉ DANS LES ANNÉES 2000, AVEC L'ARRIVÉE DE SMARTPHONES TOUJOURS PLUS PUISSANTS COMME LE BLACKBERRY ET LE NOKIA N95.

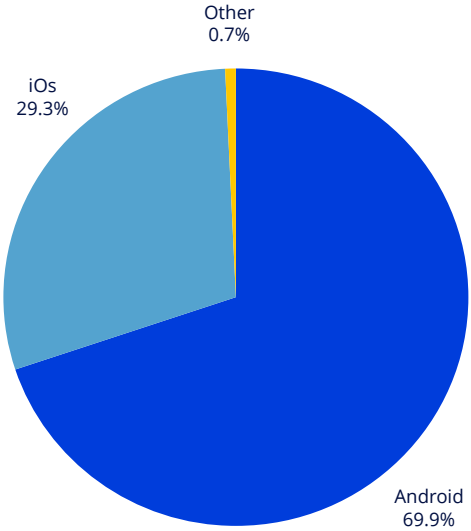
Ces appareils ont inauguré une nouvelle ère, offrant des capacités de calcul, de connectivité et d'interaction bien plus avancées que les simples téléphones mobiles de l'époque.

Cette évolution technologique a ouvert la voie à de nouvelles possibilités passionnantes, notamment la création d'applications mobiles sur mesure et capables de répondre aux besoins spécifiques des utilisateurs.

Aujourd'hui, le marché du développement mobile s'est considérablement étoffé et s'articule désormais autour de trois grands acteurs :

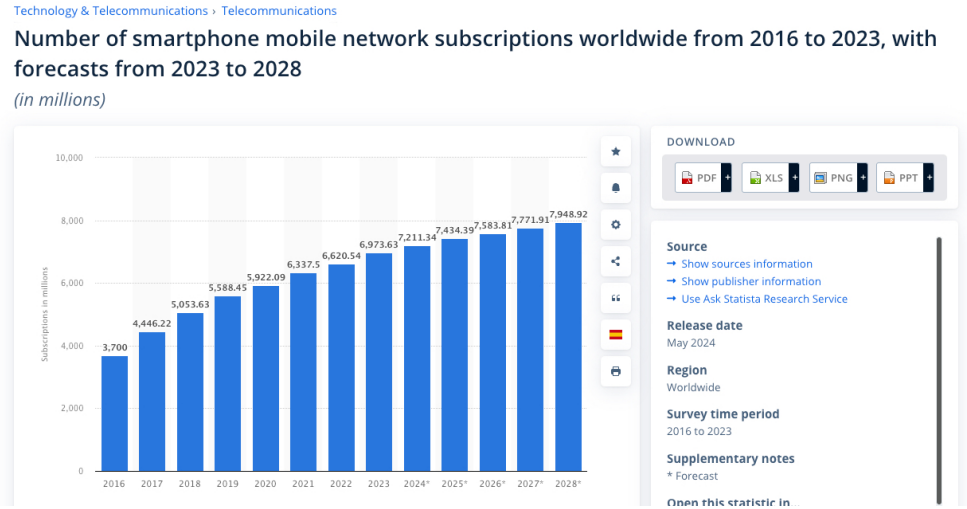
- **Android** : Le système d'exploitation open source développé par Google domine largement le marché avec quasiment 70 % de part de marché début 2024.
- **iOS** : Le système d'Apple lancé en 2007 représente 29,3% du marché début 2024.
- **HarmonyOS** : Développé par Huawei, ce système vise à offrir une alternative à ses concurrents, notamment sur les marchés où Google et Apple sont moins présents. HarmonyOS ne représente pas encore une part de marché significative mais elle est à considérer si l'on veut s'implanter sur le marché chinois.

MOBILE OPERATING SYSTEM MARKET SHARE (WORLDWIDE)



source : **https://www.mobiloud.com/blog/android-vs-ios-market-share**

01 — MARCHÉ



Le marché du développement mobile a connu une croissance phénoménale ces dernières années, portée par l'essor fulgurant des smartphones. Selon un rapport de Statista[1], le nombre d'utilisateurs de smartphones dans le monde devrait atteindre 7.4 milliards d'ici 2025, contre 3,7 milliards en 2016. Cette explosion de l'utilisation des appareils mobiles a profondément transformé les usages et les comportements des consommateurs.

Aujourd'hui, le mobile est devenu notre allié du quotidien. Les utilisateurs sont sur leur smartphone en moyenne 32 heures par semaine[2] pour des activités aussi variées que la navigation web, les réseaux sociaux, les jeux, le e-commerce ou encore comme solution de paiement dématérialisée. Mais ces smartphones sont également utilisés dans les entreprises comme outils permettant d'augmenter la productivité (en entrepôt ou en usine par exemple). Cette omniprésence du mobile dans la vie des consommateurs a contraint les entreprises à repenser complètement leur stratégie numérique, et pour certaines, à adopter une approche "mobile first".

1 Statista, "Number of smartphone users worldwide from 2016 to 2025",
https://www.statista.com/statistics/330695/number-of-smartphone-users-worldwide/
2 Vie-publique.fr, "Usages du numérique 2022",
https://www.vie-publique.fr/en-bref/288030-barometre-du-numerique-les-chiffres-cles-2022

COMMENT DÉMARRER LE **DÉVELOPPEMENT DE SON APPLICATION ?**

02

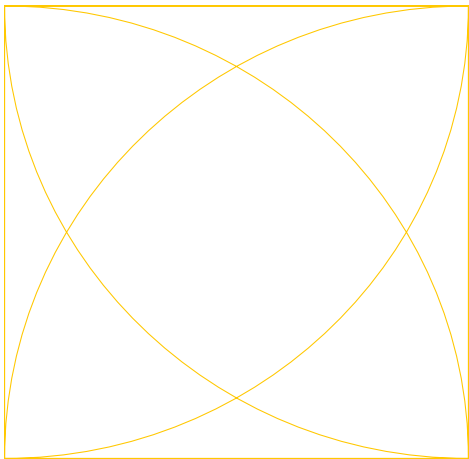
> LA PREMIÈRE ÉTAPE AVANT TOUT DÉVELOPPEMENT EST DE TESTER SON CONCEPT VIA UN POC (PROOF OF CONCEPT).

Une fois celui-ci validé, se pose alors la question de commencer le développement de votre application mobile. Choisir sa technologie, suivre les guidelines, publier sur le store, autant d'étapes pouvant apporter leur lot de complications si l'on ne connaît pas l'écosystème mobile.

De plus, le développement mobile présente des défis uniques, en particulier en ce qui concerne la diversité des formats d'écran. Le marché des appareils mobiles se caractérise par une multitude de configurations différentes, allant des smartphones compacts aux tablettes grand format, en passant par les montres connectées et les téléphones pliables. Cette fragmentation des formats impose aux développeurs une approche particulièrement rigoureuse dans la conception de leurs interfaces.

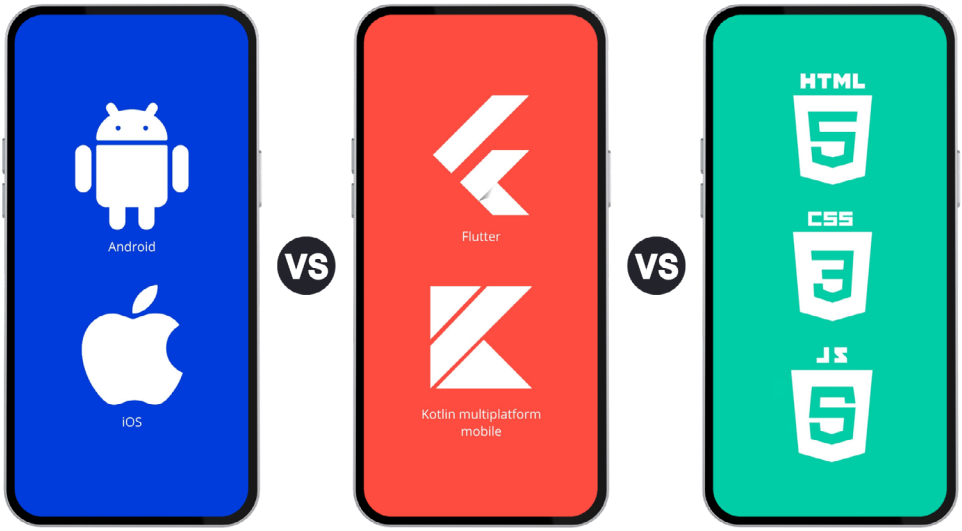
Au-delà des contraintes d'affichage, le développement mobile offre des opportunités uniques grâce à la multitude de capteurs intégrés dans les appareils modernes : GPS pour la géolocalisation, caméras performantes, accéléromètres pour détecter les mouvements, modules NFC pour les communications de proximité, etc.

Cette diversité de capteurs ouvre la voie à des fonctionnalités innovantes et des expériences utilisateur enrichies, mais cela peut vite devenir un casse-tête dès que l'on veut développer une application disponible chez les deux principaux OS tout en minimisant les coûts. Il faut alors bien choisir vers quelle technologie se diriger.



01 CHOISIR SA TECHNOLOGIE

Il existe désormais un large éventail de technologies et de frameworks différents pour réaliser des applications mobiles. Chez Ippon Technologies, nous classifions ces différents outils en quatre catégories.



LE DÉVELOPPEMENT NATIF

Le développement natif représente l'approche traditionnelle du développement mobile, où les applications sont créées spécifiquement pour chaque plateforme en utilisant leurs langages et outils officiels. Cette approche vous offre les meilleures performances et une expérience utilisateur optimale, mais nécessite des ressources plus importantes. Pour les smartphones/tablettes Android, ce sont les langages Java et Kotlin qui sont supportés de manière officielle par Google. Kotlin est un langage récent et expressif, privilégié par les développeurs Android.

Il est désormais le langage par défaut dans Android Studio lors du démarrage d'un nouveau projet et reste entièrement interopérable avec Java, toujours supporté mais un peu moins mis en avant dorénavant.

Côté smartphones/tablettes Apple, c'est Swift qui est le langage privilégié pour le développement d'applications iOS. Ce langage a pour but de remplacer l'Objective-C, encore supporté mais complètement délaissé par Apple et sa communauté. C'est un langage moderne qui offre un accès complet aux API natives du téléphone.

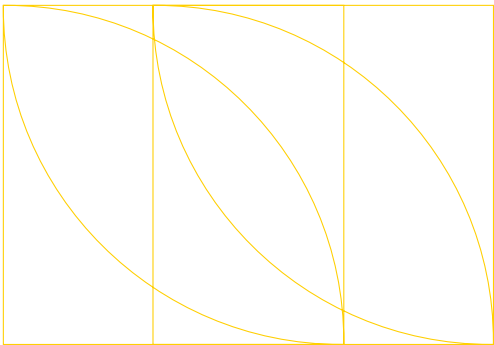
Choisir du natif implique d'avoir deux bases de code et verra vos applications potentiellement évoluer indépendamment. Néanmoins, seul le natif permet d'offrir une application au plus proche de l'expérience voulue par les développeurs des SDKs pour les différentes plateformes. Développer en natif vous permettra également d'avoir un accès plus direct aux différentes fonctionnalités software et hardware de votre téléphone (GPS, accéléromètre, appareil photo, etc).

À l'heure actuelle, une très grande majorité des applications grand public et à grande audience sont développées en natif.

LE DÉVELOPPEMENT CROSS-PLATFORM

Le développement cross-platform représente une approche stratégique dans le monde du développement mobile, permettant de créer des applications fonctionnant sur plusieurs systèmes d'exploitation avec une seule base de code. Cette approche offre des avantages significatifs en termes de coûts et d'efficacité, car elle évite la nécessité de maintenir des équipes de développement séparées pour iOS et Android. Elle permet également un déploiement plus rapide sur le marché et une maintenance simplifiée, puisque les corrections de bugs et les mises à jour peuvent être appliquées simultanément sur toutes les plateformes.

Cependant, le développement cross-platform présente aussi des limitations. Les performances peuvent être légèrement inférieures à celles des applications natives, particulièrement pour les fonctionnalités complexes ou les animations sophistiquées. L'accès à certaines fonctionnalités spécifiques à chaque plateforme peut parfois nécessiter des développements supplémentaires, et l'expérience utilisateur peut parfois manquer du "feeling" natif propre à chaque système d'exploitation.



Parmi les plus populaires, React Native, Flutter et Kotlin Multiplatform représentent la majorité des applications cross-platform :

- **React Native** : c'est un framework open-source développé par Facebook qui permet de créer des applications mobiles natives pour iOS et Android à partir du même code JavaScript. Il utilise la bibliothèque React pour construire l'interface utilisateur et offre une expérience native grâce aux composants UI natifs.
- **Flutter** : c'est un kit de développement logiciel (SDK) open-source créé par Google pour développer des applications mobiles, web et desktop à partir d'une seule base de code. Il utilise le langage Dart et propose son propre moteur de rendu, offrant ainsi des performances optimales et une apparence cohérente sur différentes plateformes.
- **Kotlin Multiplatform** : tout récemment, Kotlin Multiplatform est venu enrichir l'offre cross-platform en proposant de partager la logique métier tout en conservant des interfaces graphiques natives. Il s'agit d'une approche plus proche du bas niveau que celle des autres frameworks mentionnés, nécessitant souvent plus de travail manuel mais offrant une grande flexibilité.

Ces solutions ont gagné en maturité et sont désormais utilisées par quelques applications grand public comme Instagram (React Native), Google Pay (Flutter) ou Netflix (Kotlin Multiplatform). Elles peuvent également être un bon compromis pour des applications dont l'audience est limitée (applications internes, etc.).

ATTENTION CEPENDANT, LA PROMESSE D'UNE SEULE BASELINE DE CODE PEUT TRÈS VITE ÊTRE COMPLIQUÉE À TENIR EN FONCTION DES CHOIX DU PRODUIT OU DE LA TECHNOLOGIE.

Ces frameworks disposant de la possibilité d'interagir avec le code natif peuvent vite être amenés à justement maintenir du code customisé sur les deux plateformes, multipliant ainsi les différentes sources de code et augmentant la complexité.

LE DÉVELOPPEMENT HYBRIDE

Le développement hybride représente une approche “intermédiaire" entre le web mobile et les applications natives, permettant aux développeurs de créer des applications mobiles en utilisant des technologies web standards (HTML, CSS et JavaScript). Cette approche est particulièrement attrayante pour les équipes ayant une forte expertise en développement web, car elle permet de réutiliser les compétences existantes.

Les frameworks comme Ionic ou Apache Cordova permettent d'encapsuler votre application web dans une WebView qui est en quelque sorte un navigateur intégré à l'application mobile.

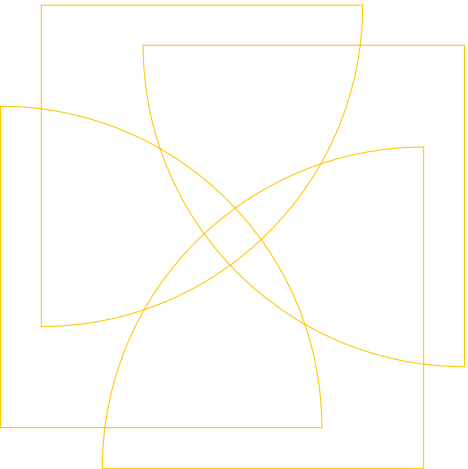
Ces outils vous donnent accès aux fonctionnalités natives du téléphone via un système de plugins, permettant par exemple d'utiliser l'appareil photo ou le GPS. Ionic, qui s'appuie sur Cordova, propose en plus une collection complète de composants d'interface préconçus qui reproduisent l'apparence native de chaque plateforme.

Le choix du développement hybride implique nécessairement des compromis en termes de performances, l'application s'exécutant dans une WebView plutôt que directement sur le système natif. Un manque de fluidité peut très vite se faire ressentir.

Cette approche est particulièrement adaptée pour les applications d'entreprise internes, les prototypes rapides ou les applications nécessitant une mise sur le marché accélérée. De grandes entreprises comme MarketWatch ou Pacifica ont fait le choix de l'hybride pour certaines de leurs applications, démontrant que cette approche peut répondre à des besoins réels tout en optimisant les coûts de développement.

LES PWAS (ET TWAS)

Il convient également de mentionner une autre approche liée au développement mobile : les Progressive Web Apps (PWA). Les PWA sont des applications web survitaminées qui offrent un compromis intéressant entre les applications web et natives. En respectant un certain nombre de guidelines officielles, elles permettent aux utilisateurs d'accéder à une expérience applicative complète directement depuis leur navigateur, tout en bénéficiant de fonctionnalités traditionnellement réservés aux applications natives, comme le fonctionnement hors-ligne ou les notifications push. Les PWA garantissent une expérience utilisateur relativement fluide et rapide tout en réduisant significativement les coûts de développement et de maintenance. Ce n'est toutefois pas nécessairement une solution pérenne sur Apple. En effet, avec l'entrée en vigueur du Digital Market Act, Apple a tout simplement interdit l'encapsulation de PWA dans une application conteneur. Ces applications peuvent donc se voir refuser la validation sur l'App Store.



Les TWA constituent quant à elles une "surcouche" spécifique à Android permettant de publier une Progressive Web App directement sur le Google Play Store. Cette technologie encapsule une PWA dans une application Android native minimale, offrant ainsi une distribution via les stores traditionnels tout en conservant les avantages du web moderne. Les TWA permettent d'accéder à davantage d'API natives et de fonctionnalités système, tout en maintenant une base de code unique et "web-first".

FAIRE LE BON CHOIX

Les fonctionnalités

Il n'y a pas de vérité absolue lorsqu'il s'agit de choisir sa technologie. Le premier critère repose sur les fonctionnalités attendues de votre application : Doit-elle accéder au hardware du téléphone de manière répétée ? Existe-t-il des contraintes fortes de performance et/ou de temps réel ? Alors le développement natif peut vous faire gagner du temps et de l'argent sur le long terme.

Les compétences disponibles

Le deuxième critère se fera sur les compétences disponibles au sein de votre entreprise et/ou de votre équipe : Ai-je des ressources en natif ? Ai-je des développeurs Dart sous la main ? Ai-je déjà une app web qui pourrait être suffisante si elle était encapsulée dans une application ? Dans de nombreuses organisations, de nombreux développeurs web possèdent davantage d'expérience en React.

La montée en compétence sur React Native peut alors se faire assez facilement tout en prenant bien en considération qu'une formation sur l'écosystème mobile reste indispensable.

La pérennité de l'outil

Le troisième critère reste aussi la pérennité de la technologie ainsi que sa courbe d'apprentissage. Prenez le temps d'évaluer la communauté de chaque outil pour déterminer si une technologie semble pérenne, maintenue et active. Flutter par exemple possède une courbe d'apprentissage relativement faible avec une grande communauté. C'est d'ailleurs celle-ci qui a récemment forké le projet Flutter[1] par manque de réactivité des développeurs Google, mainteneurs officiels du framework. Par ailleurs, Kotlin Multiplatform peut s'appuyer sur la communauté Kotlin/Android pour assurer son développement.

Le coût total sur le long terme

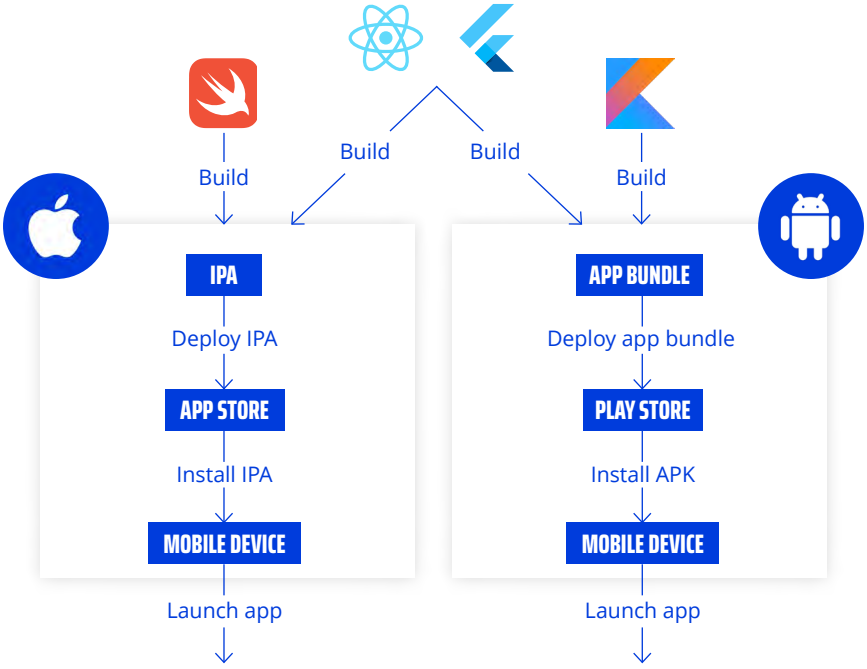
Le quatrième critère, qui est finalement le plus important si aucun des critères précédents n'a forcé un choix, reste le coût. Au-delà du coût de développement existe aussi le coût de montée en compétence et le coût de maintien. Du point de vue financier et stratégique, plusieurs facteurs clés doivent guider la décision. Le coût total de possession varie significativement :
le développement natif implique des investissements plus élevés en ressources humaines mais offre un meilleur contrôle à long terme.
Les solutions cross-platform comme React Native ou Flutter permettent de réduire les coûts de développement de 30 à 40% en moyenne, tout en maintenant un niveau de qualité encourageant. La rapidité de mise sur le marché, déterminante dans un environnement concurrentiel, favorise les approches cross-platform ou PWA mais remet en question les performances. Enfin, la disponibilité des talents sur le marché doit être considérée : React Native bénéficie du plus grand vivier de développeurs, suivi par Flutter en forte croissance, tandis que les développeurs natifs restent un peu plus rares.

1 Fork Flutter, **https://github.com/join-the-flock/flock**

02 CHOISIR SA CI/CD

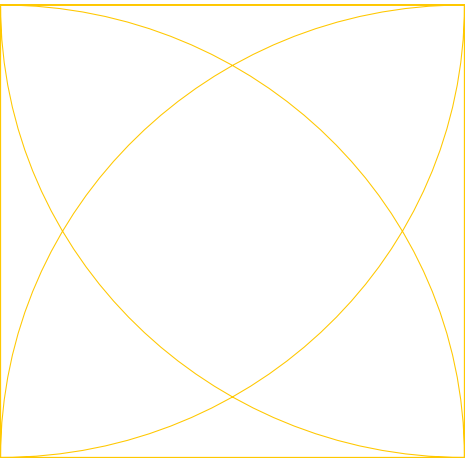
L'INDUSTRIALISATION DES APPLICATIONS MOBILES EST DEVENUE AUJOURD'HUI UNE NÉCESSITÉ POUR LES ENTREPRISES DÉSIREUSES DE RESTER EN COMPÉTITION SUR UN MARCHÉ DYNAMIQUE.

L'automatisation des processus de construction, test et déploiement de votre application devient à ce titre rapidement un enjeu fondamental.



Elle présente en effet de nombreux avantages. D'une part, elle permet de systématiser les tests et de garantir la stabilité de votre application, donc d'assurer la qualité. D'autre part, elle permet d'accélérer ces processus pour réduire vos délais de mise sur le marché. Elle permet également d'être plus efficace en limitant les coûts liés au travail humain. Les outils d'industrialisation permettent finalement d'avoir une meilleure visibilité sur l'état du projet et de suivre ses performances pour mieux optimiser les coûts.

Cette industrialisation est souvent représentée par le sigle CI/CD (Continuous Integration/Continuous Deployment) et il en existe deux grands types. Selon votre maturité, vos ressources et vos contraintes, l'une ou l'autre des approches peut s'avérer plus adaptée.



CI/CD MANAGÉES

Les CI/CD dites managées offrent des plateformes clés en main afin d'automatiser vos processus. La simplicité est la priorité de ce type de CI/CD, grâce à une configuration et des outils de mise en œuvre rapide. Les fournisseurs y assurent la sécurité et la disponibilité de vos infrastructures, vous payez à l'usage et n'avez donc pas à investir dans du matériel ou une équipe dédiée. Il existe même aujourd'hui des outils spécifiques mobile tels que Bitrise (dont Ippon est partenaire), CodeMagic, Buddy ou encore Xcode Cloud.

CI/CD SELF-HOSTÉES

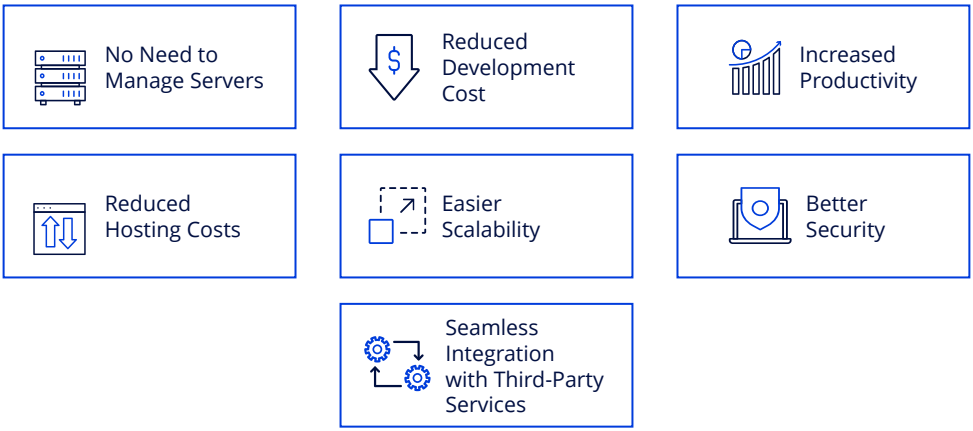
Les CI/CD dites self-hostées offrent quant à elles une plus grande flexibilité mais nécessitent une gestion plus poussée de votre part. Vous pouvez ainsi adapter plus finement vos processus à vos besoins spécifiques et choisissez les technologies s'intégrant parfaitement avec vos outils. Attention cependant, malgré un contrôle total sur votre infrastructure et vos processus, vous devez gérer l'installation, la maintenance ainsi que la sécurité sur vos serveurs. Si vous publiez une application iOS, gardez à l'idée qu'une machine tournant sous macOS est indispensable pour construire votre archive dans l'optique de publier sur le store.

03 ENVISAGER UN MBAAS

LES ENTREPRISES DÉVELOPPANT DES APPLICATIONS MOBILES FONT FACE À DE NOMBREUX DÉFIS TECHNIQUES, NOTAMMENT EN CE QUI CONCERNE LA GESTION DU BACKEND.

La mise en place d'une infrastructure robuste, sécurisée et évolutive nécessite des compétences et des ressources importantes, souvent hors de portée des équipes mobiles.

BENEFITS OF MBAAS PLATFORM



Pour répondre à cette problématique, les grands acteurs de l'informatique proposent désormais des solutions de Backend as a Service (MBaaS - Mobile Backend as a Service). Ces plateformes permettent aux développeurs de se concentrer sur l'expérience utilisateur de leur application mobile, en déléguant la gestion du backend à des experts.
Parmi les principales fonctionnalités offertes par ces solutions MBaaS, on retrouve :

- L'authentification des utilisateurs, permettant de sécuriser l'accès aux données et aux fonctionnalités de l'application. Les grands fournisseurs proposent généralement des systèmes d'authentification avancés (SSO - authentification unique, authentification par réseau social, biométrie, etc.)
- La gestion de bases de données, offrant des services de stockage et de synchronisation des données, adaptés aux usages mobiles. Cela évite aux équipes de développement de gérer directement les infrastructures de stockage.
- L'implémentation de fonctions serverless (Lambda / Cloud Functions / ...) pour le traitement de tâches côté serveur, sans avoir à déployer et maintenir des serveurs dédiés.
- Le suivi et l'analyse des usages de l'application (tracking), grâce à l'intégration d'outils de monitoring et de reporting.
- La distribution et la publication de l'application sur les stores, simplifiée par des processus automatisés.

Parmi les principaux acteurs proposant des solutions MBaaS, on peut citer **Google Firebase**, **Microsoft Azure App Center** ou encore **AWS Amplify.** D'autres entreprises comme AppWrite et Supabase peuvent constituer des alternatives plus intéressantes financièrement. Ces plateformes permettent aux équipes mobiles de se concentrer sur leur cœur de métier, tout en bénéficiant d'une infrastructure backend évolutive, sécurisée et facilement intégrable.

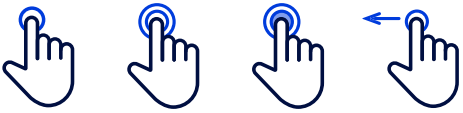
04 SUIVRE LES GUIDELINES OS

LES SPÉCIFICITÉS DE CHAQUE SYSTÈME D'EXPLOITATION MOBILE DOIVENT ÊTRE PRISES EN COMPTE.

Les interfaces utilisateur, les APIs, les performances et les contraintes diffèrent entre Android et iOS. Votre application doit savoir s'adapter à ces différences pour offrir la meilleure expérience possible, quel que soit le terminal utilisé.

Cela nécessite une expertise et une conception adaptée, plutôt que de simplement "porter" votre application d'un système à l'autre. Une approche "cross-platform" mal maîtrisée peut en effet dégrader significativement l'expérience utilisateur car Les habitudes des utilisateurs d'iPhone ne sont pas les mêmes que celles des utilisateurs de smartphones Android.

INTERACTIONS UTILISATEUR

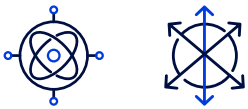


Press | Double Press | Long Press | Drag

Scroll | Swipe | Pinch

HARDWARE

Localisation | Camera | Biométrie



Gyroscope | Accéléromètre

IL VOUS FAUT ALORS BIEN ÉTUDIER LES HABITUDES DE VOS UTILISATEURS :

Souhaitez-vous offrir une expérience unifiée ou vous adapter à chaque plateforme ? La première solution offre un gain de temps et d'argent, mais peut parfois manquer d'efficacité si elle ne convient pas aux utilisateurs de telle ou telle plateforme. La deuxième solution quant à elle permet réellement de s'adapter à chaque profil mais est beaucoup plus coûteuse à implémenter.

05

RGPD - LE RÈGLEMENT GÉNÉRAL DE PROTECTION DES DONNÉES

DANS LE DÉVELOPPEMENT D'APPLICATIONS MOBILES, LA CONFORMITÉ AU RÈGLEMENT GÉNÉRAL SUR LA PROTECTION DES DONNÉES (RGPD) EST BIEN ENTENDUE OBLIGATOIRE.

Cette réglementation européenne impose une approche rigoureuse dans la gestion des données personnelles des utilisateurs. Le principe fondamental du "Privacy by Design" exige d'intégrer la protection des données dès la conception de l'application, plutôt que de l'ajouter comme une fonctionnalité supplémentaire par la suite.

Concrètement, le développement d'une application conforme nécessite l'implémentation de mécanismes spécifiques. La gestion du consentement doit être transparente et granulaire, permettant aux utilisateurs de contrôler précisément les données qu'ils partagent. Les développeurs doivent également prévoir des fonctionnalités permettant aux utilisateurs d'exercer leurs droits : accès à leurs données, possibilité de modification et option de suppression complète. Des solutions comme Didomi permettent de garantir la conformité au RGPD et de démontrer votre engagement envers la protection des données personnelles de vos utilisateurs.

Les particularités des applications mobiles, notamment l'accès aux capteurs du téléphone (géolocalisation, appareil photo, microphone), nécessitent une attention particulière. La collecte de données doit se limiter au strict nécessaire pour le fonctionnement de l'application, et chaque accès aux fonctionnalités sensibles du téléphone doit être justifié et approuvé explicitement par l'utilisateur. Cette approche, bien que contraignante, renforce la confiance des utilisateurs et devient un véritable avantage concurrentiel.

06 ACCESSIBILITÉ

AFIN D'AMÉLIORER LA PERFORMANCE ET LA QUALITÉ DE VOTRE APPLICATION MOBILE, IL EST ESSENTIEL DE GARANTIR SON ACCESSIBILITÉ AU PLUS GRAND NOMBRE, Y COMPRIS AUX PERSONNES EN SITUATION DE HANDICAP.

Cela implique de respecter des standards comme le RGAA (Référentiel Général d'Amélioration de l'Accessibilité) et le RAAM (Référentiel d'Accessibilité pour les Applications Mobiles).

Au-delà de la conformité réglementaire, rendre votre application accessible implique également de proposer des alternatives textuelles et un ordre de lecture cohérent à tous les éléments visuels. Cela garantit que les contenus soient parfaitement compréhensibles par les utilisateurs malvoyants ou non-voyants, grâce à la lecture d'écran par exemple.

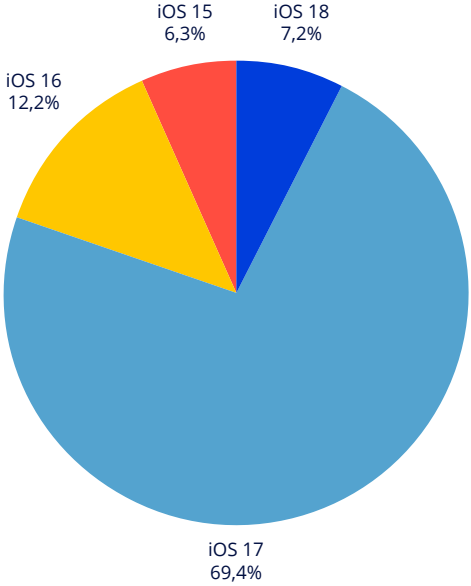
Enfin, une navigation intuitive et fluide est indispensable pour garantir une expérience utilisateur agréable à tous. L'organisation des fonctionnalités, l'arborescence des menus et l'ergonomie générale de l'interface doivent faciliter l'utilisation de l'application, y compris pour les personnes en situation de handicap.

En résumé, rendre votre application mobile accessible est un levier majeur pour en améliorer la performance globale et la qualité, tout en s'assurant qu'elle puisse être utilisée par le plus grand nombre.

07 SUPPORT DES ANCIENNES VERSIONS

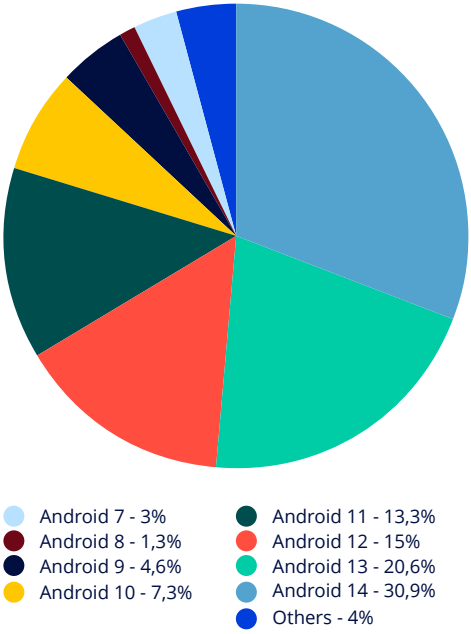
La gestion des différentes versions de systèmes d'exploitation constitue un défi majeur dans le cycle de vie d'une application mobile. Votre application doit pouvoir fonctionner sur une large gamme d'appareils et de versions d'OS, avec leurs spécificités respectives. Cette hétérogénéité, bien que difficile à gérer, constitue une opportunité pour toucher un public plus large et fidéliser vos utilisateurs à long terme.

IOS VERSION USAGE



source : **https://iosref.com/ios-usage** (arrêté au 1er octobre 2024)

ANDROID API LEVELS



source : **https://apilevels.com/** (arrêté au 1er septembre 2024)

LE CHOIX DES VERSIONS MINIMALES À SUPPORTER EST UNE DÉCISION STRATÉGIQUE QUI IMPACTERA DIRECTEMENT LE SUCCÈS DE VOTRE APPLICATION.

Pour prendre cette décision, analysez la répartition des versions d'OS utilisées par votre public cible et intégrez des tests de compatibilité dans votre processus de développement. Utilisez de vrais appareils et tenez compte des retours des utilisateurs, qui révèlent souvent des problèmes de compatibilité en conditions réelles.

L'anticipation des nouvelles versions d'OS est tout aussi importante que le support des anciennes. Profitez des versions bêta fournies par les constructeurs pour adapter votre application aux changements à venir et soyez transparent avec vos utilisateurs sur les versions supportées et leurs limitations. Pensez également à consulter les notes de mise à jour lors des sorties des versions majeures des systèmes d'exploitation.

DISTRIBUER SON APPLICATION

03

LES MODES DE DISTRIBUTION DE VOS APPLICATIONS PEUVENT DIFFÉRER SELON LE BESOIN QUE VOUS EN AVEZ.

Le déploiement d'une application au sein de votre entreprise ne se fait pas nécessairement de la même manière que sa diffusion au grand public.
La distribution d'une application mobile représente une étape importante dans son cycle de vie, nécessitant une stratégie bien pensée et une compréhension approfondie des différents canaux disponibles. Les stores officiels (l'App Store d'Apple et le Google Play Store de Google) constituent la vitrine principale pour atteindre vos utilisateurs. Au-delà des stores officiels, d'autres options de diffusion s'offrent aux entreprises selon la cible.

01

DÉPLOYEZ VOTRE APPLICATION AU SEIN D'UNE ENTREPRISE

DE NOMBREUSES ENTREPRISES UTILISENT AUJOURD'HUI DES FLOTTES DE TERMINAUX MOBILES AU SEIN DE LEURS ORGANISATIONS.

Que ce soit pour améliorer la productivité ou renforcer la sécurité des téléphones en circulation, de nombreux outils sont disponibles pour mieux gérer ces smartphones. Parmi ces solutions de Mobile Device Management (MDM) on peut notamment citer Intune, SOTI ou encore Appaloosa.

Les MDM offrent de nombreux avantages aux organisations cherchant à déployer et gérer les applications sur les appareils de leurs collaborateurs. Ces solutions permettent de mettre en place une sécurité renforcée en appliquant des politiques cohérentes à l'échelle de l'ensemble du parc d'appareils. La protection des données de l'entreprise est considérablement renforcée grâce à des fonctionnalités avancées de chiffrement et de règles d'accès strictes.

En cas de perte ou de vol d'un appareil, les administrateurs peuvent rapidement réagir en localisant, verrouillant ou effaçant à distance les données sensibles, minimisant ainsi les risques de fuite d'informations.

Le déploiement et la gestion des applications se trouvent également grandement simplifiés grâce à ces plateformes de MDM. Les équipes IT peuvent désormais déployer et mettre à jour les applications de manière centralisée et automatisée sur l'ensemble des terminaux de l'entreprise. Cette capacité à pré-configurer les applications et à les distribuer en masse garantit une homogénéité des installations et des mises à jour sur tous les appareils, réduisant le temps et les ressources nécessaires à la maintenance du parc mobile.

02

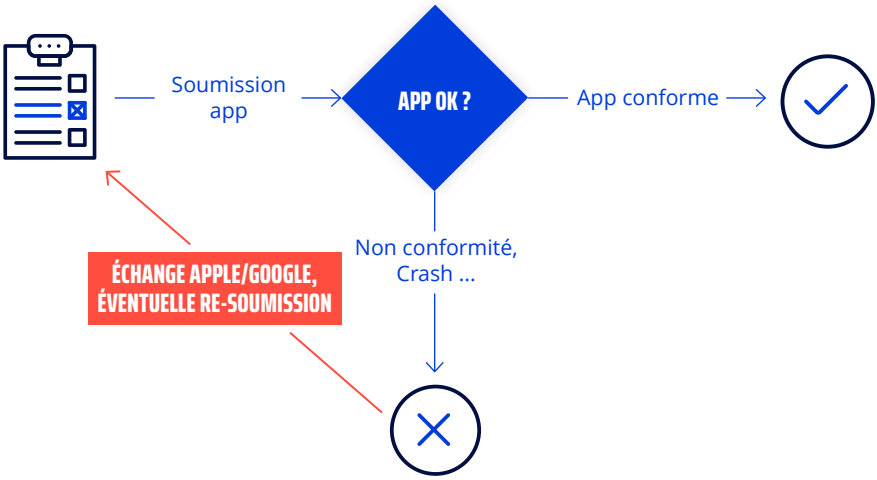
DÉPLOYEZ VOTRE APPLICATION AUPRÈS DU GRAND PUBLIC

L'IMPORTANCE DE LA PUBLICATION SUR LES STORES EST DÉTERMINANTE DANS LE DÉVELOPPEMENT MOBILE.

Les magasins d'applications comme l'App Store pour iOS et Google Play Store pour Android sont les principaux canaux de distribution pour les applications mobiles. Ils offrent une visibilité immédiate à des millions d'utilisateurs potentiels, ce qui est essentiel pour atteindre un large public. Ces plateformes vérifient les applications avant leur mise en ligne, renforçant ainsi la confiance des utilisateurs dans la sécurité de leurs téléchargements. Cette vérification contribue à créer un environnement sûr pour les consommateurs, ce qui est vital pour le succès d'une application.

La signature de votre application est obligatoire avant sa publication sur un store. Elle garantit que l'application provient bien de vous et n'a pas été altérée pendant le téléchargement, assurant ainsi l'intégrité du code. Ce mécanisme de sécurité empêche toute modification malveillante après la compilation, protégeant à la fois les développeurs et les utilisateurs. Sur certains systèmes, comme Android, la signature est même nécessaire pour installer l'application hors des stores officiels, offrant une flexibilité supplémentaire aux développeurs.

De plus, elle permet de vérifier que les mises à jour proviennent bien du développeur original, maintenant la cohérence et la qualité de l'expérience utilisateur au fil du temps. Certaines fonctionnalités avancées, comme les services en arrière-plan, nécessitent également une signature valide pour s'intégrer correctement au système d'exploitation.



Attention cependant, la publication sur les stores peut s'avérer compliquée. Il faut respecter de nombreuses guidelines et être conforme aux deux stores principaux (App Store, Play Store). Cela ne se fait pas sans mal car les deux stores fonctionnent différemment.

Chez Apple, une vérification manuelle est réalisée et les guidelines peuvent sembler plus contraignantes. Par exemple, il est obligatoire de fournir la méthode d'authentification "Sign in with apple" si vous désirez utiliser n'importe quel autre mécanisme d'authentification par réseau social. Par ailleurs, si votre application permet la création de compte utilisateur, vous êtes dans l'obligation d'en proposer la suppression depuis l'application. Enfin, un dernier exemple est que chaque application doit respecter visuellement les "Human Interface Guidelines"[1].

Pour tout manquement aux contraintes établies par Apple, vous verrez votre version de l'application refusée. Notez que cette validation n'a pas de durée fixe et dépend de la disponibilité des équipes d'Apple. Cela peut prendre une demi-journée comme plusieurs jours.

Côté Google, la validation est automatisée par des robots et peut donc s'avérer plus souple et plus rapide (quelques heures). Cependant, il existe des règles strictes notamment concernant la politique de confidentialité, les contenus sensibles et la "Data Safety Section". Il s'agit d'un questionnaire qui précise les données collectées par l'application et leur usage exact.

1 Human Interface Guidelines, **https://developer.apple.com/design/human-interface-guidelines**

PASSER
À L'ÉCHELLE

04

Avoir publié une première version de votre application est un jalon important, mais le plus délicat reste de réussir à la faire vivre et à la faire grandir. Pour cela, il est nécessaire de savoir industrialiser votre application. Cela passe par l'adoption de processus, de concepts et d'outils permettant de passer à l'échelle, tout en répondant aux besoins des utilisateurs et en évitant l'accumulation de dette technique pouvant mener à une refonte précipitée.

01 — TESTS

LORSQU'ON PARLE DE "QUALITÉ", LES TESTS SONT LA PREMIÈRE CHOSE QUI VIENT À L'ESPRIT.

Outre les tests bien connus que sont les tests unitaires et d'intégration, qui font partie intégrante du cycle de développement d'une application, il est intéressant de noter que d'autres types de tests peuvent intervenir pour s'assurer qu'une application est qualitative à plus grande échelle.

À ce titre, les tests UI automatisés permettent de vérifier le bon fonctionnement des fonctionnalités métier de l'application. Ils simulent les actions des utilisateurs dans des conditions réelles ou quasi-réelles, permettant de vérifier que l'interface réagit correctement et que les comportements globaux sont conformes aux attentes. Ces tests sont essentiels pour s'assurer que l'expérience utilisateur est cohérente et fluide sur les différents appareils et configurations. Dans les faits, ils permettent également de s'assurer que des scénarios entiers de parcours utilisateurs sont cohérents et fonctionnels.

Le Snapshot testing quant à lui permet de s'assurer de la conformité d'une interface utilisateur, d'un écran, par rapport à un attendu déterminé par l'UX/UI et par rapport à un existant. Cette approche permet de détecter rapidement les changements visuels non intentionnels et d'assurer la cohérence de l'apparence de l'application au fil du temps.

Une culture du test doit être développée au sein de l'équipe. Chaque nouvelle fonctionnalité devrait être accompagnée de ses tests correspondants. L'adoption de pratiques comme le Test-Driven Development (TDD) ou le Behavior-Driven Development (BDD) peut faciliter cette transition vers une approche de qualité systématique. Les tests manuels exploratoires gardent néanmoins leur importance, particulièrement pour évaluer l'expérience utilisateur et découvrir des cas d'utilisation imprévus. La combinaison de ces tests automatisés et d'une exploration manuelle permet ainsi de construire une stratégie de test complète, capable d'accompagner efficacement la croissance de l'application.

02 SÉCURITÉ DE L'APPLICATION MOBILE

LA SÉCURITÉ EST UN ASPECT PRIMORDIAL À PRENDRE EN COMPTE LORS DU DÉVELOPPEMENT D'UNE APPLICATION MOBILE.

Les applications manipulent de nombreuses données sensibles, comme les informations personnelles des utilisateurs, les contenus propriétaires ou les accès à des fonctionnalités critiques. Toute faille de sécurité pourrait avoir de graves conséquences, tant pour les utilisateurs que pour la réputation de l'entreprise.

Les bonnes pratiques de sécurité doivent être intégrées à chaque étape du processus de développement. Des ressources comme le Mobile OWASP[1] peuvent aider à définir un standard mobile pour la sécurité. Loin d'être une contrainte, la sécurité doit être perçue comme un levier pour gagner la confiance des utilisateurs.

Cependant, le travail ne s'arrête pas là. Il est important de régulièrement auditer l'application, aussi bien fonctionnellement que techniquement, afin de déceler d'éventuelles failles. Un audit externe par un expert impartial apporte plusieurs bénéfices :

- Il permet d'avoir un regard neuf et une connaissance à jour des dernières tendances et standards de sécurité mobile.
- L'expert identifiera des axes d'amélioration que l'équipe interne aurait pu ignorer ou sous-estimer.
- Ses recommandations renforceront la protection globale de l'application et réduiront les risques.

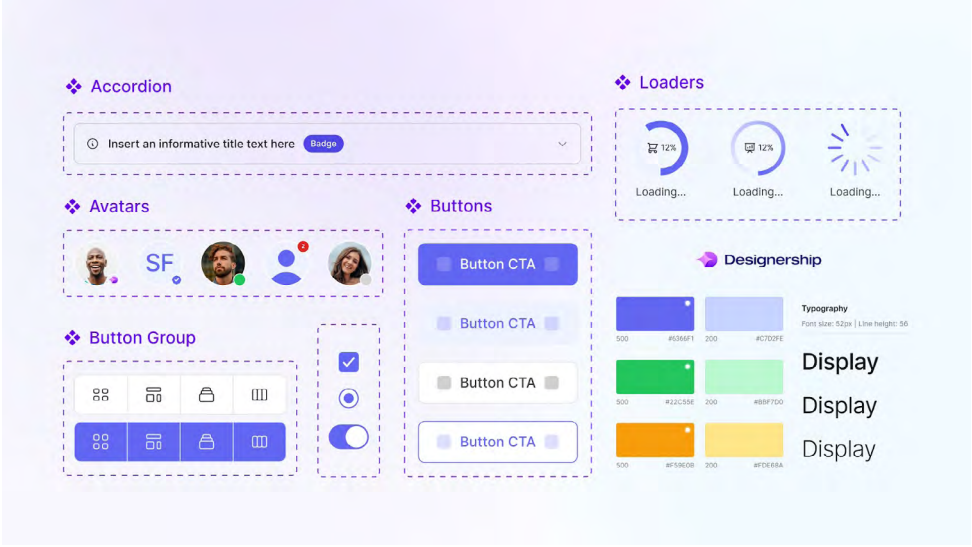
Des outils comme MobSF[2] peuvent également être utilisés pour analyser statiquement l'archive de l'application et détecter de potentielles vulnérabilités.

1 OWASP, **https://owasp.org/www-project-mobile-app-security/**
2 MobSF, **https://github.com/MobSF/Mobile-Security-Framework-MobSF**

03 DESIGN SYSTEM

La cohérence et l'uniformité de l'expérience utilisateur sont des enjeux décisifs pour les applications mobiles.
Face à la multiplicité des fonctionnalités et des écrans, l'interface utilisateur risque de se fragmenter et de perdre sa cohérence. C'est particulièrement vrai dans les grandes organisations qui gèrent plusieurs applications mobiles simultanément. La mise en place d'un design system devient alors non plus une option, mais une nécessité stratégique.

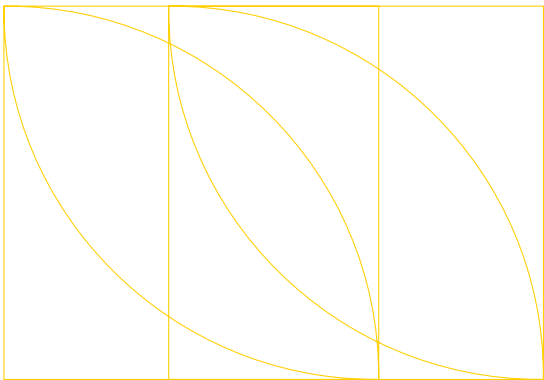
Un design system va bien au-delà d'une simple bibliothèque de composants. Il constitue un véritable langage commun entre les différentes équipes de l'organisation, englobant les éléments graphiques, les règles d'interaction et les principes de conception. Dans un contexte multi-applications, il unifie l'expérience utilisateur sur l'ensemble des produits de l'entreprise. Cela renforce l'identité de marque et améliore la reconnaissance immédiate par les utilisateurs.



source : **https://www.thedesignership.com/blog/benefits-of-good-design-system**

Au sein de grandes entreprises comptant de nombreuses équipes “Projet”, il accélère considérablement le développement de nouvelles applications ou fonctionnalités. Les équipes n'ont plus besoin de réinventer la roue à chaque nouveau projet : elles peuvent s'appuyer sur une base solide de composants éprouvés et documentés, réduisant considérablement les coûts de développement et de maintenance.

La collaboration entre les équipes s'en trouve également transformée. Designers, développeurs et product managers parlent le même langage, utilisant des références communes et des processus standardisés. Dans un environnement où plusieurs équipes travaillent en parallèle sur différentes applications, cette harmonisation des pratiques est précieuse. Elle facilite la mobilité des équipes entre les projets et accélère l'intégration de nouveaux collaborateurs.



04 — USAGES & MESURES

LE SUIVI ET L'ANALYSE DES USAGES DE VOTRE APPLICATION MOBILE SONT DES ÉLÉMENTS CLÉS POUR EN ASSURER LE SUCCÈS ET L'AMÉLIORATION CONTINUE.

Sans une compréhension claire du comportement de vos utilisateurs, il devient difficile de prendre des décisions stratégiques pertinentes et d'optimiser efficacement votre produit.

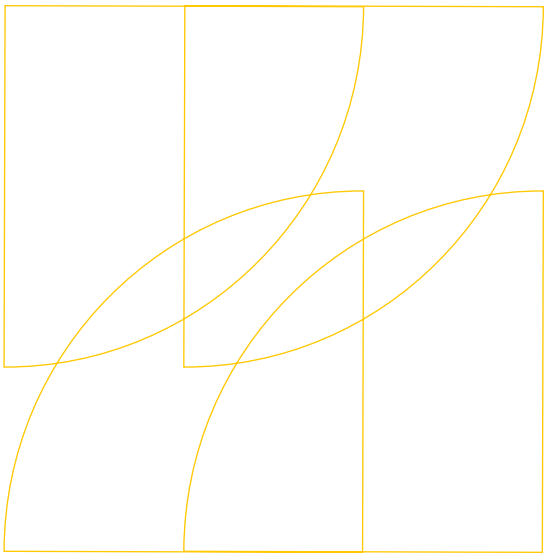
C'est pourquoi il est essentiel de définir dès le départ un plan de tracking exhaustif pour votre application. Celui-ci doit identifier les métriques les plus pertinentes à suivre, en fonction de vos objectifs métiers et des attentes de vos utilisateurs.

Parmi les indicateurs classiques à suivre, on peut notamment citer :

- Le nombre d'utilisateurs actifs et leur évolution dans le temps.
- Les pages et écrans les plus consultés, ainsi que leur temps de consultation moyen.
- Les taux de conversion des différentes actions clés, comme les inscriptions, les achats ou les partages.
- Les sources d'acquisition des nouveaux utilisateurs.
- Les principaux points de friction et d'abandon dans le parcours utilisateur.

En complément de ces métriques quantitatives, il est également important de recueillir des données qualitatives, par le biais d'études utilisateurs, de sondages ou de tests d'utilisabilité. Cela permet de mieux comprendre les motivations, les frustrations et les attentes réelles de vos utilisateurs.
Une fois les indicateurs définis, la mise en place des outils de tracking appropriés est l'étape suivante. Des outils spécialisés comme Firebase Analytics, Matomo, ou Amplitude pourront être mis en place pour recueillir ces informations.

À côté de cela, des outils peuvent être mis en place pour monitorer votre application. La ou l'analytics se focalise sur l'utilisateur et le produit, une mise en place d'une bonne politique d'observabilité permet de surveiller l'état de son application. Avec des outils comme Datadog ou Sentry, on peut alors exploiter des données permettant d'évaluer de manière fiable les performances de l'application, ses entrées et sorties, ses crashs ou encore détecter des potentielles attaques. Mettre en place une politique de monitoring permet de s'appuyer sur des données claires pour résoudre d'éventuels problèmes ou même de les anticiper.



05 — ASO

L'ASO (APP STORE OPTIMISATION) REPRÉSENTE L'ENSEMBLE DES TECHNIQUES ET DES PRATIQUES VISANT À AMÉLIORER LA VISIBILITÉ ET LE RÉFÉRENCEMENT SUR LES STORES.

Dans un marché saturé comptant des millions d'applications, l'ASO est devenu un élément clé pour se démarquer et atteindre son public ciblé. Cette pratique s'apparente au SEO (Search Engine Optimisation), mais avec des spécificités propres à l'écosystème mobile.

L'ASO repose sur plusieurs piliers fondamentaux. Les métadonnées de l'application, comprenant le titre, la description, la catégorie et les mots-clés, doivent être soigneusement choisies pour correspondre aux recherches des utilisateurs. Les éléments visuels, comme les icônes, les captures d'écran et les vidéos de présentation jouent également un rôle capital dans la conversion des visiteurs en utilisateurs. La performance de l'application, mesurée à travers les notes, les avis et les statistiques d'utilisation, influence directement son classement dans les stores. Les techniques peuvent différer sur chaque store. Par exemple, le Play Store de Google permet d'afficher plus de captures d'écran que l'App Store d'Apple (8 contre 5 chez Apple).

Une stratégie ASO efficace nécessite un suivi régulier ainsi que des ajustements constants. L'analyse des performances, la veille concurrentielle et l'adaptation aux évolutions des algorithmes des stores sont essentielles pour maintenir et améliorer la visibilité de votre application.

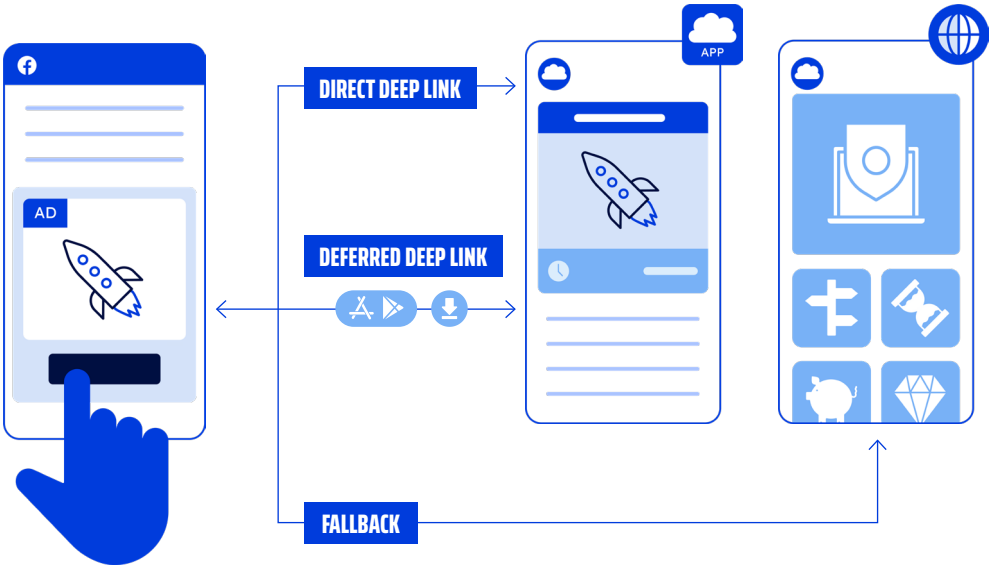
06 DEEPLINKS

LES DEEPLINKS REPRÉSENTENT UNE TECHNOLOGIE PROPRE À L'ÉCOSYSTÈME MOBILE.

Ils permettent de créer des liens directs vers des contenus spécifiques au sein d'une application. Contrairement aux liens web classiques, les deeplinks peuvent diriger l'utilisateur vers une page ou une fonctionnalité précise de l'application, comme un produit particulier, un article ou une section spécifique.

Dans une perspective d'analytics, les deeplinks constituent un outil précieux pour comprendre et optimiser le parcours utilisateur. Ils sont également un vecteur important dans l'augmentation de l'engagement utilisateur et dans la rétention de celui-ci. Plus vous proposez d'options et de liens "deeplinks" renvoyant au sein de votre propre application, plus les utilisateurs seront actifs sur celle-ci. On peut citer en exemple les applications E-commerce qui permettent de rediriger du navigateur Web vers l'application au clic d'un article dans le moteur de recherche du navigateur. Selon un article de Google[1], Shopee, une entreprise de e-commerce leader en Asie, a constaté dès 2021 une augmentation significative du taux de conversion de ses transactions.

1 Google, A deep-link case study, **https://www.thinkwithgoogle.com/marketing-strategies/app-and-mobile/shopee-deep-link-case-study/**



Dans une perspective d'analytics, les deeplinks constituent un outil précieux pour comprendre et optimiser le parcours utilisateur. Ils sont également un vecteur important dans l'augmentation de l'engagement utilisateur et dans la rétention de celui-ci. Plus vous proposez d'options et de liens "deeplinks" renvoyant au sein de votre propre application, plus les utilisateurs seront actifs sur celle-ci. On peut citer en exemple les applications E-commerce qui permettent de rediriger du navigateur Web vers l'application au clic d'un article dans le moteur de recherche du navigateur. Selon un article de Google[1], Shopee, une entreprise de e-commerce leader en Asie, a constaté dès 2021 une augmentation significative du taux de conversion de ses transactions.

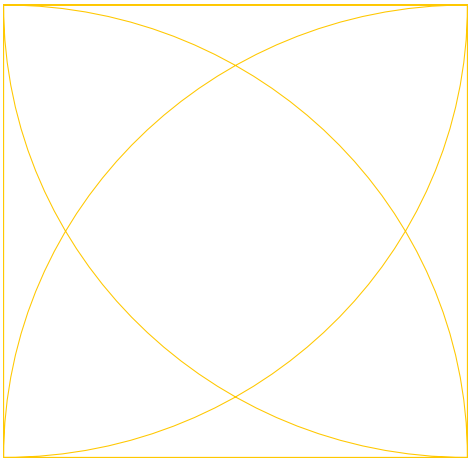
1 Google, A deep-link case study, **https://www.thinkwithgoogle.com/marketing-strategies/app-and-mobile/shopee-deep-link-case-study/**

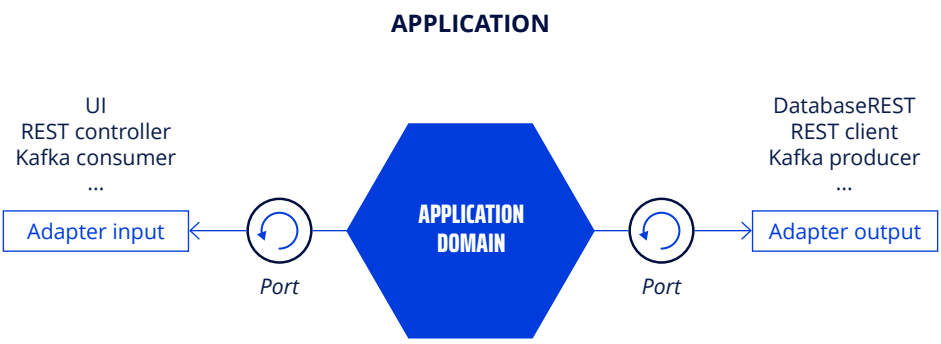
07 — CRAFT

LA CONCEPTION TECHNIQUE D'UNE APPLICATION MOBILE DOIT ÊTRE PENSÉE AVEC SOIN AFIN DE GARANTIR SA PÉRENNITÉ ET SA CAPACITÉ À S'ADAPTER AUX ÉVOLUTIONS DU MARCHÉ.

Adopter les bonnes pratiques du développement mobile, et d'une manière plus générale, celles du développement logiciel de qualité, est essentiel pour atteindre cet objectif. Adopter une approche Craft peut grandement contribuer à rendre l'application souple et modulaire.

Cette façon de penser et de développer offre de nombreux bénéfices : cette méthode facilite la maintenance du code, accélère les itérations de développement et renforce la réactivité face aux changements. L'adoption d'une architecture technique appropriée contribuera grandement à la scalabilité de votre application, aussi bien en termes de facilité de maintenance qu'en rapidité d'évolution. En séparant clairement les responsabilités et en minimisant les dépendances, il devient bien plus aisé d'ajouter de nouvelles fonctionnalités, de corriger des bugs ou d'expérimenter de nouvelles solutions sans impacter l'ensemble de l'application.





Dependency injection

Une autre pratique essentielle pour gagner en agilité est l'utilisation de feature flags. Cela consiste à encapsuler de nouvelles fonctionnalités dans des blocs de code conditionnels, activables ou désactivables à distance ou selon certaines conditions. Une fois ce système en place, vous pouvez déployer de nouvelles versions de l'application de manière progressive, tester des fonctionnalités en production sur un sous-ensemble d'utilisateurs, ou encore désactiver rapidement une fonctionnalité problématique.

08 ÊTRE MOINS DÉPENDANT DES STORES

La validation des applications par les stores peut effectivement ralentir le déploiement de nouvelles fonctionnalités. Cependant, plusieurs approches permettent de contourner ou d'optimiser ce processus :

La *remote configuration* représente une solution puissante pour gérer dynamiquement le comportement de votre application. En implémentant un système de configuration à distance, vous pouvez activer ou désactiver des fonctionnalités, modifier des paramètres ou adapter le contenu sans nécessiter une nouvelle version de l'application. Des outils comme Firebase Remote Config ou des solutions personnalisées permettent cette flexibilité. Une utilisation concrète des “Remote configuration” est le feature flipping. Cette technique permet de déployer du code en production tout en contrôlant son activation. Les fonctionnalités peuvent ainsi être testées sur un groupe restreint d'utilisateurs ou activées progressivement, réduisant les risques et permettant une réactivité accrue. Le code est déjà présent dans l'application, mais son activation est pilotée à distance.

Les solutions de mise à jour à chaud (hot update) comme CodePush/AppCenter pour React Native (qui sera désactivé fin 2025) ou ShoreBird pour Flutter offrent une approche encore plus directe. Ces outils permettent de déployer des modifications de code directement aux utilisateurs sans passer par les stores, tant que ces changements ne concernent pas le code natif ou les ressources packagées. C'est une méthode particulièrement efficace pour corriger des bugs, optimiser les performances ou actualiser du contenu. Cependant, leur utilisation doit respecter les directives des stores : Apple impose notamment des restrictions strictes sur les modifications de code après publication.

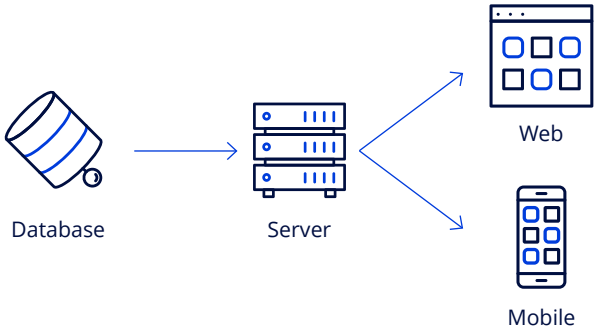
Le server-driven UI représente une approche plus avancée où l'interface utilisateur elle-même est pilotée par le serveur. Cette méthode permet de modifier substantiellement l'expérience utilisateur sans mise à jour de l'application. Bien que plus complexe à mettre en œuvre au départ, elle offre une flexibilité maximale pour l'évolution de votre application.

ET SI ON PASSAIT MOBILE FIRST ?

INDUSTRIALISER SON APPLICATION EST DÉJÀ UNE RÉUSSITE IMPORTANTE.

Certaines entreprises vont encore plus loin. Depuis l'essor des smartphones, on observe depuis plusieurs années des produits ayant une approche Mobile First. Celle-ci consiste à prioriser le développement pour les appareils mobiles, en concevant d'abord l'expérience utilisateur pour le mobile avant de la décliner sur d'autres supports. Selon Google, 61% des utilisateurs sont moins susceptibles de revenir sur un site web non optimisé pour le mobile 8. Il est donc essentiel pour les entreprises d'offrir une expérience adaptée aux appareils mobiles.

Ces contraintes sont nombreuses et impactent profondément le développement d'applications mobiles. Comparé à une application web classique, une application mobile doit tenir compte de contraintes telles que la taille réduite de l'écran, la mobilité de l'utilisateur, l'accès aux capteurs du téléphone (GPS, caméra, accéléromètre, etc.), la gestion de la batterie ou encore les différences entre iOS et Android. Les technologies utilisées pour le développement mobile, comme Swift, Kotlin, React Native et Flutter, permettent de relever ces défis. permettent de relever ces défis et d'offrir une expérience utilisateur adaptée aux appareils mobiles.



Sur le plan financier, le marché du mobile représente un énorme potentiel de revenus. En 2022, les revenus mondiaux des applications mobiles ont atteint 167 milliards de dollars, en hausse de 19% par rapport à 2021[1]. Les entreprises qui ont adopté une stratégie mobile first, comme WhatsApp, Instagram ou TikTok, figurent parmi les acteurs les plus performants de ce marché. Par exemple, TikTok a généré 16,1 milliards de dollars de revenus en 2023 (soit une augmentation de 67%) et est devenu l'une des applications les plus lucratives au monde[2].

1 Sensor Tower, "The State of Mobile 2024", **https://sensortower.com/state-of-mobile-2024**
2 Sensor Tower, "TikTok Revenue and Usage Statistics (2023)", **https://www.businessofapps.com/data/tik-tok-statistics/**

CRÉDITS PHOTOS

Photo de **Alin Gavriliuc** sur Unsplash - p31
Photo de **Balazs Ketyi** sur Unsplash - p36
Photo de **AS Photography** de Pixabay - p24
Photo de **Pexels** de Pixabay - p26
Photo de **NASA** sur Unsplash - p45
Photos générées par **Midjourney** - p12,14,20
Photos sur **canva.com** - p21,22,39

ippon

CONTACTEZ-NOUS !

fr.ippon.tech
blog.ippon.fr

contact@ippon.fr
+33 1 46 12 48 48
@ippontech