الجـــــــــــــــــــمـهـوريـــــة الجـــــــــــــزائـريـة الديمـــــــقـراطيـة الـشـــــــــعبيـة

REPUBLIQUE ALGERIENNE DEMOCRATIQUE ET POPULAIRE

وزارة التـــــــــــــــــــــــــــعـليـم العـــــــالي و البـــــــــحث العـــــلـمـــي

Ministère de l'Enseignement Supérieur et de la Recherche Scientifique

جـــــــــــــــامعة أبي بكـــر بـلـقـايد – تـلمســـــــــــــان –

Université Aboubakr Belkaïd – Tlemcen –

Faculté de TECHNOLOGIE

MEMOIRE

Présenté pour l'obtention du **diplôme** de **MASTER**

En : Télécommunication

Spécialité : Réseaux Mobiles et Service de Télécommunications

Par : Midoun Younes Omar

Dib Mohamed Ilyes

Sujet

Développement d'une application Android dans le domaine pharmacologique

Soutenu publiquement, le 24 / 05 / 2016, devant le jury composé de :

Mr. MERZOUGUI. R	**M.C à l'Université de Tlemcen**	**Président**
Mr. BAHRI. A	**M.C à l'Université de Tlemcen**	**Examinateur**

Remerciements

Nous remercions en premier lieu le bon dieu qui nous a donné la force et le courage pour terminer ce travail.

Le travail présenté dans ce projet de fin d'études a été effectué dans le cadre de la préparation du diplôme du master en télécommunications spécialité « Réseaux mobiles et services de télécommunications » à l'Université Abou Bekr Belkaïd – Tlemcen.

Nous souhaitons adresser nos remerciements les plus sincères aux personnes qui nous ont apporté leur aide et qui ont contribué à l'élaboration de ce mémoire ainsi qu'à la réussite de cette formidable année universitaire.

La première personne que nous tenons à remercier est notre encadrant Monsieur HADJILA MOURAD, pour l'orientation, la confiance et la patience qui ont constitué un apport considérable sans lequel ce travail n'aurait pas pu être mené au bon port. Qu'il trouve dans ce travail un hommage vivant à sa haute personnalité.

Que Monsieur MERZOUGUI Maitre de Conférences à l'Université Abou Bekr Belkaïd – Tlemcen, qui nous fait l'honneur de présider ce jury, ainsi que le membre de ce jury Moniseur BAHRI Maitre de Conférences à l'Université Abou Bekr Belkaïd – Tlemcen, veuillent bien accepter nos sincères remerciements.

Nous tenons à remercier également Monsieur DIB DJILALI, Pharmacien de profession, pour l'honneur qu'il nous a fait en acceptant de nous fournir des informations précieuse. Sa disponibilité, sa gentillesse ont été primordiales. Qu'il puisse trouver dans ce travail le témoignage de notre sincère gratitude et de notre profond respect.

Dédicace

A tous ceux qui m'ont soutenu tout au long de ce projet.

A mes chers parents et mes frères, que nulle dédicace ne puisse exprimer ce qu'on leurs doit, pour leurs bienveillance, leur affectation et leur soutien morale durant l'élaboration de ce travail,

en témoignage de mon profond amour et nos sincères reconnaissances pour les efforts qu'ils ont consenti pour l'accomplissement de mes études,

je leur dédie ce modeste travail.

Omar

Dédicace

Je dédie ce travail :

A mes parents,

Merci d'avoir toujours cru en moi. Merci pour votre amour, votre soutien, vos encouragements et votre aide, dans les bons comme dans les mauvais moments.

A mes fréres Eisam, Imrane, Mourad et ma soeur Ikram

Merci pour votre joie de vivre et vos conseils si précieux.

A tous mes amis

Pour les bons moments passés et à venir. Pour votre précieuse amitié.

ILYES

Sommaire

Préface

Dans un monde actif ou l'évolution peut être perçu à l'œil nu, la motivation de développer des moyens plus performants et plus efficaces de communication et d'échange d'informations devient de plus en plus importante. Cette motivation nous pousse à une révolution portant à bien l'utilisation de tous les appareils embarqués pour des besoins personnels et quotidiens.

Nous remarquons ces dernières années un développement exponentiel des appareils mobiles qui sont répandus comme une traînée de poudre dans le monde en développement et révolutionnant le domaine des communications. Dans cet environnement, ce n'est plus les anciens téléphones portables qui font l'unanimité mais c'est les Smartphones qui apparaissent pour rompre avec les opinions de ses prédécesseurs et donner une autre dimension à cette technologie tout en incorporant de nouveaux apports et services à la téléphonie mobile et en attirant la clientèle grâce à sa facilité d'utilisation révolutionnaire et attractive.

En partant de l'idée d'optimiser l'utilité d'un Smartphone, le développement d'une application a vu le jour. Après avoir fait quelques études sur le domaine pharmacologique tout en établissant nos propres statistiques, on a conclu qu'il y a un nombre croissant de personnes atteintes de plusieurs maladies, donc soumises à une poly médication (médicaments pour le cœur, le diabète, l'hypertension ...). Le constat amer auquel sont confrontés les médecins est : Comment faire cohabiter plusieurs médicaments simultanément chez le même malade en évitant les interactions qui pourraient porter préjudice à l'intégralité physique du malade ?

Partant sur une base de données scientifique ayant pour corolaire l'évitement des effets nocifs des interactions médicamenteuses, il nous a paru qu'il était possible de développer une application qui pourrait rendre un service énorme à la prescription médicale et un support de vulgarisation de l'information au malade.

L'utilité des Smartphones va modifier d'une manière significative les gestions de la vie des malades. Ces appareils mobiles joueront le rôle de lien entre le médicament et le médecin, et le médicament et le malade.

Ils fournissent aux utilisateurs individuels et aux communautés un accès précieux à toute une série de services d'information à des fins personnelles.

Introduction :

Un médicament est d'abord le fruit de la recherche pharmaceutique. En prenant compte les recherches et le développement qu'a connu, on pourra dire qu'on a fait un grand pas vers l'avant. Autant précisé que la découverte de plusieurs maladies a conduit à une croissance énorme du nombre de médicaments et cela nous confronte à de gigantesques défis.

Le pharmacien a le devoir de rechercher les possibles interactions médicamenteuses issues des ordonnances qu'il délivre, mais aussi de veiller à ne pas provoquer une interaction lors de la délivrance d'un produit conseillé.

Pour l'aider dans cette démarche, il dispose d'ouvrages, de banques de données et de logiciels. Cependant, ces outils sont le plus souvent fondés sur des données théoriques. Les publications sur les interactions médicamenteuses sont nombreuses mais leurs pertinences cliniques et biologiques ne sont pas toujours évidentes. Hors, seules les associations médicamenteuses comportant réellement des conséquences significatives sont à cibler préférentiellement.

Alors tous les médicaments ne font pas bon ménage entre eux. On parle d'*interaction médicamenteuse* lorsque la prise d'une substance modifie l'effet d'un ou plusieurs autres principes actifs présents au même moment dans l'organisme.

Prendre un médicament n'est jamais anodin. En prendre plusieurs en même temps encore moins, est-ce une lapalissade pour tout le monde ? Pas certain du tout, dès lors que l'on prend plus de deux médicaments en même temps, les conditions sont alors créées pour que leur efficacité et leur toxicité soient potentiellement modifiées, mais pas obligatoirement.

Alors est ce bien si de faire une combinaison de trois, quatre, voire beaucoup plus de médicaments ? Quelle attitude, quelle conduite adopter si l'on prend plusieurs médicaments ? Quels sont les types d'interactions qui existent ? Et la question principale est: Qu'est ce qu'un médicament, quel est la définition exacte d'une interaction médicamenteuse ?

I. Qu'est-ce qu'un médicament

Le mot *médicament* provient du latin « medicamentum » signifiant remède. Aujourd'hui, le médicament est soumis à une définition rigoureuse. Les médicaments sont des produits destinés à prévenir les maladies, à les traiter, voire à les déceler. Ils peuvent en combattre la cause - c'est le cas des antibiotiques contre les infections bactériennes - ou en atténuer les manifestations, comme les antalgiques utilisés pour soulager la douleur [1].

Les médicaments contiennent généralement des substances - ou principes actifs, dont l'effet s'étend à l'ensemble de l'organisme. Il est possible de moduler leur absorption et leur durée d'action en modifiant leur structure chimique, ou en choisissant la forme adéquate. Il existe plusieurs formes de médicaments :

- Les formes orales : comprimé, gélule, sirop…
- Les formes pour application externe : pommade, crème, gel, poudre, patch…
- Les formes pour le nez, les oreilles ou les yeux : solution auriculaire, collyre…
- Les formes injectables : injection, perfusion.

II. Classification des médicaments

Il existe plus d'une dizaine de milliers de médicaments. Chaque médicament est utilisé dans un but précis et par des spécialités médicales différentes. Il y a de nombreuses façons de classifier les médicaments. Les deux plus importantes sont:

- **Classement par DCI (dénomination commune internationale) :** Un médicament est classé selon son (ou ses) principes actifs. Ce type de classification permet de retrouver un médicament dans n'importe quel pays du monde et quel que soit le nom de marque qu'il porte. On cite à titre d'exemple: *Dafalgan®, Efferalgan®, Doliprane®, Claradol®, Dolko®, Dolotec®,Geluprane®*. La dénomination commune internationale pour tous ces médicaments c'est le *Paracétamol* .

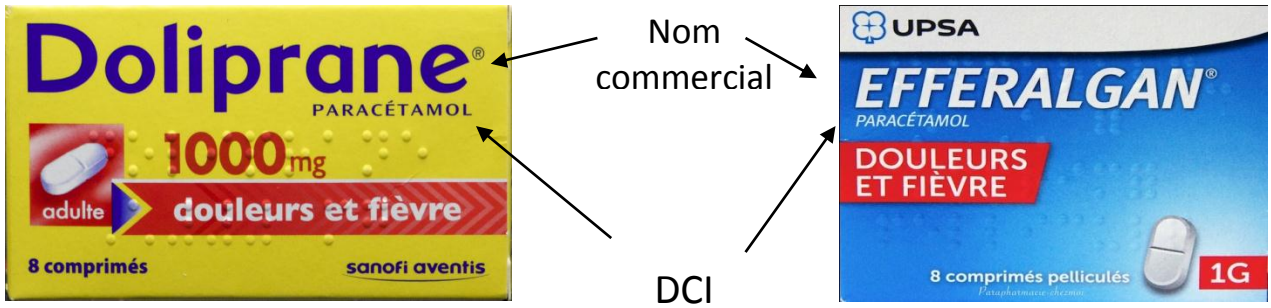


Fig. I.1- Présentation de la dénomination commune internationale (DCI).

• **Classement par action thérapeutique :** On appelle cela les "Familles pharmaco-thérapeutiques". Alors une famille peut regrouper plusieurs principes actifs, voici quelques exemples :

Famille thérapeutique	Action	Principe actif
Antalgiques	Calme et supprime la douleur due à une pathologie	Paracetamol, Morphine, Codéine
Anti-Infectieux	Permet de combattre une infection	Les antibiotiques, Les antiviraux et les vaccins
Anti-inflammatoires non stéroïdiens	Diminution des effets de l'inflammation	Acide acétylsalicylique (Aspégic), Diclofénac potassique , sodique

Tableau I.1- Quelques exemples sur les classes thérapeutiques.

III. Les interactions médicamenteuses

1. Définition

Par définition, une interaction médicamenteuse « IM » est une modification qualitative ou quantitative, des effets d'un médicament par un autre médicament, aliment, ou boisson. L'interaction peut également avoir lieu avec des agents chimiques de l'environnement. Cependant, pour ce travail, nous ne retiendrons que les interactions ayant lieu entre médicaments. Certaines interactions médicamenteuses ont des *conséquences cliniques* graves, d'autres n'ont que des effets sans critère de gravite et/ou sont parfois *asymptomatiques*[2].

Quand on parle de l'addition de plusieurs médicaments, il faudrait précisez la molécule mère (ou bien le principe actif) de chacun d'eux parce que c'est elle qui détermine sa position vis à vis à l'interaction. Ce n'est pas tous les médicaments qui sont concernés par ces interactions parce qu'on sait bien que ces médicaments sont bénéfiques, mais ils agissent parfois en contradiction les uns par rapport aux autres. Ces contradictions sont répertoriées en quatre niveaux :

1. Le niveau le plus grave est la « **contre-indication** ». Celle-ci revêt un caractère absolu, et ne doit pas être transgressée. Elle peut avoir un effet toxique qui entre dans ce que l'on appelle l'iatrogénie (conséquence négative due à un traitement ou un acte médical).

2. L'« **association déconseillée** » doit être le plus souvent évitée. Si les deux médicaments sont nécessaires, et après examen approfondi du rapport bénéfice/risque, une surveillance étroite du patient s'impose.

3. Le cas le plus fréquent est la « **précaution d'emploi** ». L'association est possible dès lors que sont respectées, notamment en début de traitement, les recommandations simples permettant d'éviter la survenue de l'interaction : adaptation des doses, renforcement de la surveillance clinique, biologique, électrocardiogramme…

4. Au quatrième niveau, il n'existe pas de recommandation pratique. Pourtant, le risque d'interaction médicamenteuse existe, et correspond le plus souvent à une **addition d'effets indésirables**.
Il revient au médecin d'évaluer l'opportunité de l'association de médicaments.

2. Conséquences pharmacologiques des interactions médicamenteuses

Les conséquences pharmacologiques des interactions médicamenteuses sont des modifications quantitatives d'un ou de plusieurs effets (thérapeutiques ou indésirables) d'un ou des médicaments de l'association. Si nous considérons un effet déterminé, elles affectent soit son intensité, soit sa durée, soit les deux paramètres simultanément. On peut compter trois conséquences :

La synergie :

L'action d'un médicament peut être augmentée en rapidité, en intensité, en durée, par l'administration simultanée d'un autre médicament possédant une activité pharmacologique qualitativement identique.

Il existe trois types de synergie :

- La synergie additive complète : l'action observée est égale à la somme des deux actions partielles.

- La synergie partielle : l'action observée est inférieure à la somme des deux actions partielles.

- La synergie potentialisatrice : l'action observée est supérieure à la somme des deux actions partielles[3].

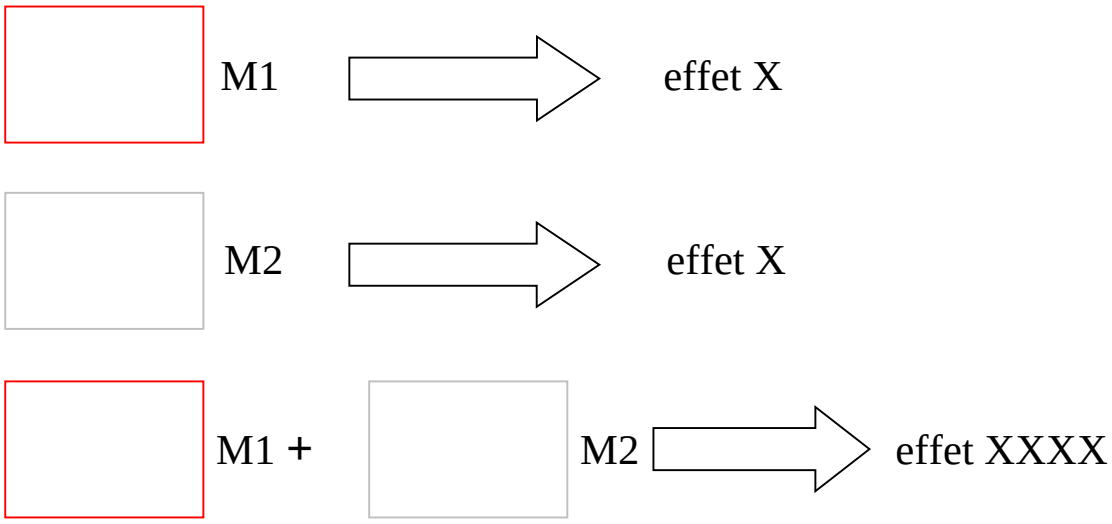


Fig. I.2- Mécanisme de la synergie.

Les IM synergiques sont très fréquemment observées. Exemple :

- Lorsque que l'on prescrit deux médicaments antihypertenseurs, le risque *d'hypotension artérielle orthostatique* est naturellement plus fréquent.
- Un antihypertenseur avec un antidépresseur ; le risque d'hypotension artérielle orthostatique est majoré.

La potentialisation : la somme des effets des deux médicaments est supérieure à leur simple addition.

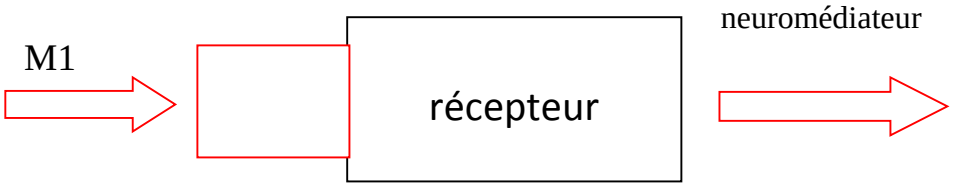
L'antagonisme : On parle d'antagonisme de deux substances, d'activité pharmacologique identique ou différente, lorsque l'une diminue ou même annule les effets de l'autre.

A .L'antagonisme total

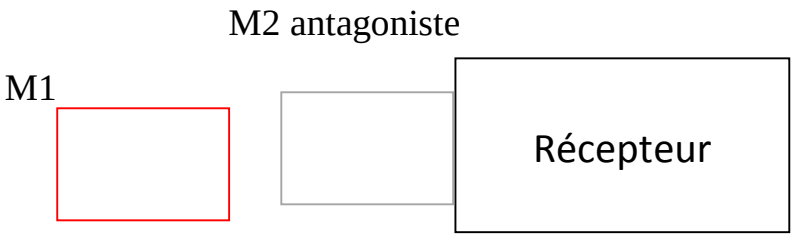
On parle d'antagonisme total lorsque les effets des deux substances s'annulent totalement.

B .L'antagonisme partiel

On parle d'antagonisme partiel lorsque l'effet global de l'association est inférieur à celui de l'un ou de l'autre de ces deux constituants pris isolément[4].



La fixation de M1 sur le récepteur entraine la libération du *neuromédiateur* .

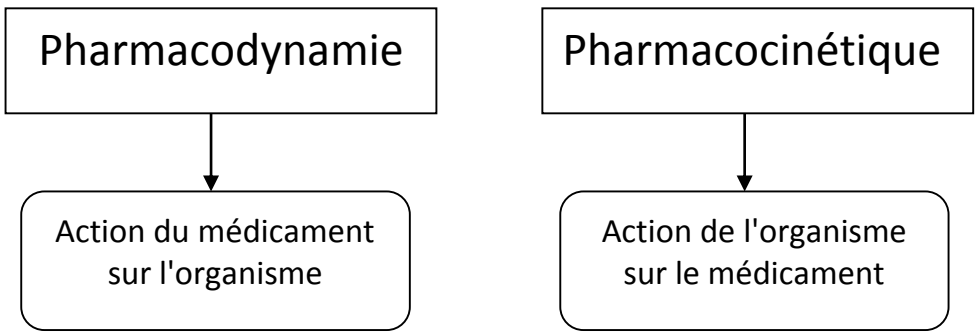


M1 ne pouvant plus se fixer sur le récepteur, occupé par M2, il n'ya plus de libération de neuromédiateur.

Fig. I.3- Mécanisme d'antagonisme.

3. Deux grands types d'interactions médicamenteuses

Les mécanismes des interactions médicamenteuses sont multiples, mais ils peuvent être classés en deux groupes : Les interactions pharmacocinétiques et les interactions pharmacodynamiques.



Interactions d'ordre pharmacocinétique

C'est la modification de la relation entre la dose administrée et la concentration du médicament actif. La pharmacocinétique a pour but d'étudier le devenir d'un médicament dans l'organisme. Ce type d'interaction existe dès l'instant où sont présents dans l'organisme deux ou plus de deux médicaments simultanément, voire consécutivement. Elles peuvent survenir à toutes les étapes du parcours d'un médicament dans l'organisme : résorption, distribution, métabolisme ou élimination.

Absorption / Résorption :

L'absorption est le passage du principe actif du médicament dans la circulation générale à partir de son lieu d'administration. Elle est influencée par plusieurs facteurs dont les principaux sont la voie d'administration, la forme pharmaceutique, la nature du médicament, ses propriétés de dissolution et l'état du site d'absorption [5].

a. Distribution :

La distribution est la répartition du principe actif du médicament depuis son entrée dans la circulation générale jusqu'à son arrivée au site d'activité.

b. Métabolisme :

Le métabolisme comprend des modifications chimiques que subit une substance médicamenteuse dans l'organisme. Le résultat de ces modifications est généralement la réduction de la substance en une forme inerte, appelée *métabolite inactif***,** plus facilement éliminable. Les substances médicamenteuses ne sont cependant pas toutes inactivées par le métabolisme ; certains médicaments deviennent des *métabolites actifs***.**

La plupart des réactions métaboliques sont enzymatiques et se produisent dans le foie.

c. Élimination :

L'élimination consiste en l'excrétion de la substance médicamenteuse hors de l'organisme. Elle est assurée par divers organes : le rein – le plus important –, le foie et les poumons. Une portion de certains médicaments peut se retrouver dans la salive, la sueur ou le lait maternel [6].

Interactions d'ordre pharmacodynamiques:

La pharmacodynamique a pour but d'étudier l'effet du médicament sur la zone ciblé. Les interactions pharmacodynamiques sont relativement prévisibles en fonction des connaissances des principaux effets des médicaments concernés. Elles concernent souvent des médicaments ayant des propriétés pharmacodynamiques ou des effets indésirables communs, complémentaires ou *antagonistes* vis-à-vis d'un même système physiologique.

Un médicament peut avoir deux effets sur un autre médicament :

- Soit par augmentation des concentrations de l'un des deux (surdosage).
- Soit par une baisse des concentrations conduisant à une inefficacité (sous-dosage).

De façon générale :

Les surdosages induisent à une augmentation des effets indésirables, par franchissement du seuil de tolérance.

Un sous-dosage induit à une inhibition de l'activité du médicament qui se situe alors sous le seuil d'efficacité du médicament.

En résumé, il s'agit de modifications des concentrations dues à l'effet d'un médicament sur l'autre.

Les interactions pharmacocinétiques sont plus difficiles à prévoir que les interactions pharmacodynamiques, et c'est dans cette étape là qu'on rencontre la majorité des interactions médicamenteuses [7].

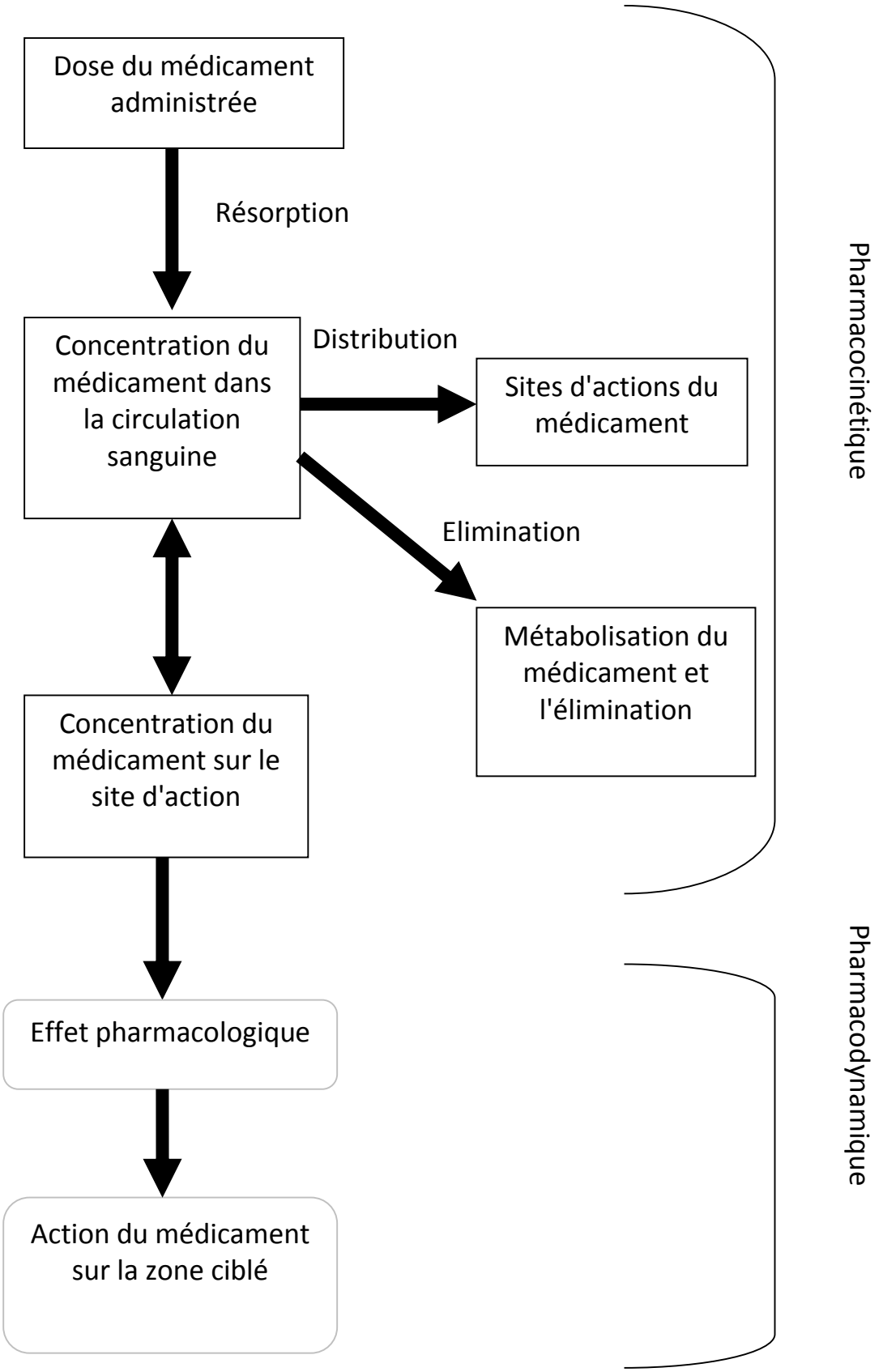


Fig. I.4- les différentes étapes parcourues par le médicament après son administration.

IV. Présentation de la problématique :

Malgré la multi-morbidité (atteinte de plusieurs maladies à la fois) et les cas d'intoxications médicamenteuse très répandue, les directives cliniques sont grande partie écrite en prenant en compte un seul traitement à la fois. L'impact cumulatif de multiples traitements est rarement considéré. Pour les personnes atteintes de plusieurs maladies, l'application des recommandations suivant les directives de traitement de maladie singulière produit un schéma thérapeutique complexe nommé (Polypharmacie) avec une potentielle combinaison des médicaments implicitement néfastes.

Selon une étude faite aux Royaume-Unis (qui présente déjà un plan médicale très efficace par rapport au reste du monde) par *TheBMJ (The British Medical Journal)*, les effets d'interactions causent en moyenne une admission de 6.5% des hospitalisations non planifiées. Prenant 4% de la capacité d'un hôpital, ils peuvent se terminer par la mort du patient principalement par hémorragie ou infection rénale. Bien que quelque de ses interactions soient prévisibles, le reste peut être prévu et évités, y compris les interactions Drug-to-Drug (entre deux médicaments) qui présente la majorité de ses cas [8].

Un chiffre pour montrer l'ampleur du phénomène en France, pays où on consomme au monde le plus de médicaments, le Ministère de la Santé évalue (mais sans citer ses sources) à 8 000 le nombre de décès annuels du seul fait des interactions médicamenteuses [9].

La grande majorité de ces décès est susceptible à la prévention par détection de ses interactions avant de prendre ces médicaments. Mais le problème qui se pose est que de telles informations existent dans un état difficilement accessible pour le grand publique, encore moins pour les personnes avec peu de connaissance dans le domaine. Bien que le pharmacien soit la source indubitable et la plus viable pour éviter un tel problème, il ne peut avoir connaissance de ce que le patient ne mentionne pas, car ce dernier n'est pas conscient de ce qui peut affecter son traitement.

V. Projet de la solution envisagée :

Pour aborder un tel problème on a opté pour le développement d'une application Android pour smartphones et tablettes, des outils exponentiellement répandus dans le monde et présents dans la majorité des demeures, qui servira à détecter de telles interactions dans le cas de polypharmacie ou juste vérifier deux médicaments.

L'application a pour but d'offrir une interface simple et ergonomique, dénouée de toute complexité et détails inutiles, qui permettra à un utilisateur ayant très peu de connaissance de connaitre les risques d'interactions entre un médicament et un autre. Il faudra donc traduire l'information brute et en extraire que le nécessaire à jauger la sévérité d'une interaction.

Vu le déploiement du nouveau Code QR (Quick Response Code) dans les médicaments récents, l'utilisation de l'application sera entièrement automatisée par un simple scan du code-barres du médicament par l'appareil photo et donnera des résultats simples et interprétables par tout le monde.

L'application permettra donc de :

- Enregistrer des traitements en cours et les garder dans une liste pour d'éventuels tests.
- Vérifier l'interaction d'un médicament avec un traitement enregistré.
- Vérifier rapidement d'éventuelles interactions entre deux médicaments.
- Scanner un médicament et en afficher les informations relatives.
- Saisi automatique des médicaments par un simple scan de son code QR.

Conclusion

Les malades ayant plusieurs traitements à suivre contenant des médicaments incompatibles entre eux doivent être conscients que la prise d'un médicament ou de tout autre produit actif n'est jamais anodine. C'est pour cela que le malade doit tenir au courant le médecin traitant de tout nouveau médicament qu'il prendra, en dehors de ses prescriptions, de même, il pourra informer les autres praticiens de santé (dentiste, chirurgien, pharmacien, autres spécialistes).

En pratique, pour les médecins, l'utilisation d'un dictionnaire Vidal® est conseillée avant de prescrire plusieurs médicaments en même temps.

Pour le moment, cela est considéré comme le seul moyen d'éviter les interactions médicamenteuses. Leur risque croît avec le nombre de médicaments sur une ordonnance.

Envisager une solution à ce problème est le but de notre travail, en offrant une possibilité de consulter les possibles risques en prenant plusieurs médicaments simultanément sans avoir recours à l'avis du médecin ou pharmacien. Mais n'empêche que ce dernier reste le moyen le plus sûr.

Introduction :

Les Smartphones et autres terminaux mobiles intelligents (tablettes, montres, télévision) utilisent des systèmes d'exploitation (ou OS : Operating System en Anglais) différents. Les principaux OS populaires utilisés dans le monde sont Android, Windows Phone, et IOS d'Apple (I-Phone OS) etc. Le système d'exploitation mobile possède clairement un impact important sur le marché de la téléphonie, d'où la présence des géants de l'industrie informatique comme Google et Apple alors que chacun d'eux développe son propre OS, Android pour le premier coté et iOS pour le second.

Des sondages et recherches nous ont permis de déterminer l'OS le plus utilisé par les usagers des Smartphone. L'Android a eu la plus grande part du marché en raison de sa nature Open Source (son code source est connu de tout le monde et modifiable par les constructeurs) il est devenu le favoris chez les utilisateurs de Smartphones ainsi que les développeurs d'applications. Ces derniers peuvent répondre aux besoins de n'importe quels usagés en créant des applications compatible avec tous les appareils exploitant l'OS Android (tablettes, Smartphones, montre, TV, lunettes).

Les applications ont pour but d'exploiter au maximum les performances d'un Smartphone, d'où l'idée d'utiliser l'objectif photographique d'un téléphone pour scanner le QR Code (Quick Response Code). Ce code est défini comme le nouveau code à barre, il a pu remplacer son prédécesseur et a redéfinis ses principales fonctions.

Ce chapitre sera divisé en trois parties distinctes : dans la première partie nous nous intéressons à l'Android en évoquant son évolution ainsi que ses caractéristiques et son architecture ; La deuxième partie introduira le nouveau code-barres " le QR Code " ainsi que son principe de fonctionnement et son utilité. Quant à troisième partie elle sera dédiée à l'environnement de développement d'une application Android.

I. Le système Android

1. Définition

Android est un système d'exploitation Open source et il est basé sur Linux. Android est conçu pour des appareils mobiles au sens large. Nullement restreint aux téléphones, il ouvre d'autres possibilités d'utilisation des tablettes, des ordinateurs portables, des bornes interactives et des smartwatches (montres intelligentes). Afin de promouvoir ce nouveau système d'exploitation ouvert, Google a su fédérer autour de lui un consortium d'une trentaine d'entreprises : l'Open Handset Alliance (OHA) créée officiellement le 5 novembre 2007. Toutes ces entreprises interviennent, plus ou moins directement, dans le marché de la téléphonie mobile puisqu'elles représentent des opérateurs téléphoniques, fabricants de semi-conducteurs, d'appareils mobiles, de logiciels [10].

Leur objectif principal était de développer des normes ouvertes pour les appareils de téléphonie mobile dans le but de trouver une solution fiable pour concurrencer Apple avec l'iOS, Microsoft et Nokia avec Windows Mobile et Research In Motion avec Blackberry OS. En octobre 2008, apparait la première version d'Android qui n'avait pas reçu de nom. Cette version s'est avérée être la version béta du système.

Implémenté dans un terminal mobile, l'Android exploite les fonctionnalités de ce dernier en utilisant des applications téléchargées dans une plateforme appelé Google Store. L'OS offre une approche unifiée au développement d'applications pour les appareils mobiles ce qui signifie que les développeurs doivent seulement développer sous Android, et leurs applications devraient être en mesure de fonctionner sur différents appareils alimentés par Android.

Le code source pour Android est disponible sous les licences de logiciels libres et gratuits, et c'est ce qui fait sa force. Un des autres atouts de l'Android c'est qu'il est personnalisable par les constructeurs et les opérateurs de téléphonie mobile. Tout en subissant plusieurs changements, le cœur du système reste commun ce qui permet une interopérabilité des applications.

Fig. II.1 : Déploiement mondiale d'Android

2. Caractéristiques d'Android

Android est un système d'exploitation puissant et supporte beaucoup de fonctionnalités. Peu d'entre elles sont énumérées ci-dessous:

Caractéristique	Description
Interface	L'OS fournit une interface intuitive et simple à utiliser
Connectivité	GSM/EDGE, IDEN, CDMA, EV-DO, UMTS, Bluetooth, Wi-Fi, LTE, NFC and WiMAX.
Stockage	SQLite une base de données relationnelle légère, pratique pour le stockage de données
Lecteur Media	H.263, H.264, MPEG-4 SP, AMR, AMR-WB, AAC, HE-AAC, AAC 5.1, MP3, MIDI, Ogg Vorbis, WAV, JPEG, PNG, GIF, and BMP
Messagerie	SMS et MMS
Navigateur Web	Utilisant les versions de Google Chrome, il supporte le HTML5 et CSS3

Multitâche	L'utilisateur peut passer d'une tâche à l'autre et en même temps diverses applications peuvent fonctionner simultanément.
widgets redimensionnables	Widgets sont redimensionnables, afin que les utilisateurs puissent les développer pour afficher plus de contenu ou de les rétrécir pour économiser l'espace
Wi-Fi Direct	Une technologie qui permet la connexion entre deux équipements en utilisant une bande passante élevée
Android Beam	Permet le partage de fichiers juste au contact physique entre deux entités possédant la technologie NFC et exploitant l'Android

Tableau II.1 - Caractéristiques de l'Android [11]

3. Les applications Android et la diversité de leur déploiement

Les applications Android sont généralement développées dans le langage Java en utilisant un kit de développement spécifique à lui. Il s'agit d'un programme qui est embarqué sur le mobile de l'utilisateur, après avoir préalablement été téléchargé de manière gratuite ou payante selon le choix de l'éditeur qui l'a créée.

Une fois téléchargée, l'application est représentée sous la forme d'une petite icône qui s'affiche sur l'écran du Smartphone. Elle s'utilise alors par la suite à l'envie simplement en appuyant sur l'icône en question, via son écran tactile.

Une application, comme son nom l'indique, remplit une fonction, une utilité précise qui peut être très variée. Elle offre une multitude d'usages différents allant du jeu en passant par l'information, l'achat, les réservations, l'écoute de musique... Ses applications sont disponibles au niveau d'une plateforme appelé Google Play Store et peuvent être téléchargées par n'importe quel utilisateur d'Android à condition qu'il suffise juste d'avoir un compte Gmail. Ceci dit, cette opération se fait dans un lien descendant, dans le cas contraire où on aura un développeur qui voudrait charger son application sur cette plateforme, une fois mis au point, cette application peut être vendue ou offerte selon le souhait de son éditeur sur des plateformes. Avec les possibilités matérielles incorporées aux terminaux (caméra, GPS, …), les applications Smartphones et Tablettes peuvent intégrer des fonctionnalités spécifiques et dédiées pour les utilisateurs, permettant ainsi d'enrichir le spectre fonctionnel et imaginer des usages non couverts jusqu'à présent. Ces applications sont présentes dans plusieurs domaines, en évoquant une fonctionnalité

quotidienne ou bien professionnelle. Nous allons citer quelques unes qui ont révolutionné l'univers de l'Android et en particulier le domaine de la santé :

iBiomed : Elle permet de conserver facilement des enregistrements et des informations sur la santé qui nous suivent partout. Elle comprend une section sur la médication, un rappel pour prendre la médication, des notes datées, un forum de discussion sur la santé.

E-Thyroïde : Une application développée par le géant de l'industrie du médicament le laboratoire "Bayer" : Cette application accompagne les patients atteints de pathologies thyroïdiennes dans le suivi de leur traitement et les aide au quotidien. E-thyroïde permet ainsi au patient de :

- Suivre des traitements
- Gérer les effets secondaires
- Gérer la vie quotidienne grâce à des conseils (alimentation, activité physique ...)
- Enregistrer ses résultats
- Regrouper ses contacts médicaux

MySugr Carnet : Cette application permet de gérer le diabète au quotidien. Elle peut enregistrer une foule de données : glycémie, glucides, activité, humeur… A l'aide de statistiques, elle va les analyser et les envoyer au médecin. Cette application est destinée aux personnes diabétiques sous insuline.

- Dans la GEOLOCALISATION, ITINERAIRES :

Google maps, Sygic, Tomtom : En utilisant différents procédés de géolocalisation, ces applications permettent :

* Améliorer l'expérience utilisateur en apportant une information localisée.

* Afficher la localisation courante d'un utilisateur et de procéder à des recherches autour de soi.

* Accompagner l'utilisateur à travers une fonctionnalité de navigation ou d'itinéraire.

4. L'évolution ascendante dans le marché

Le système de Google n'aurait pas connu un tel succès s'il était resté statique en neuf ans. C'est là que l'on remarque la puissance d'un tel OS qui a su s'adapter et répondre aux besoins des utilisateurs à chaque version majeure. Les cinq dernières versions sont les plus utilisées (Android utilise des noms de confiseries comme nom de code de la version):

- Android 4.0 (Ice Cream Sandwich)
- Android 4.1 (Jelly Bean)
- Android 4.4 (KitKat)
- Android 5.0 (Lollipop)
- Android 6.0 (Marshmallow)

A chaque nouvelle version, des modifications sont appliquées à quelques terminaux puisque parfois certains appareils ne sont pas compatibles avec les nouvelles mises à jour, voici un diagramme qui montre l'utilisation de ces versions :

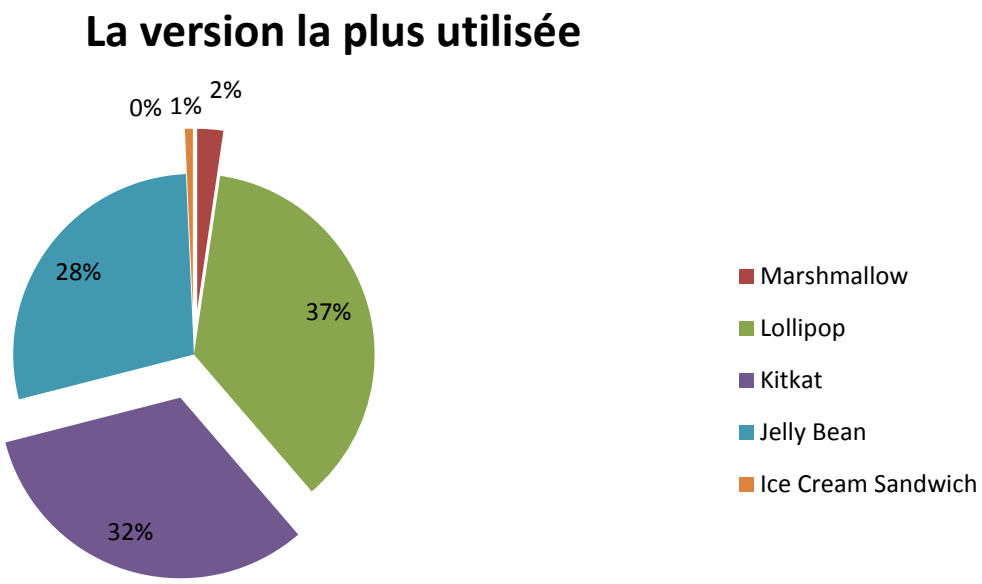


Fig. II.2–Répartition des versions Android

Android a eu un début modeste en 2008, mais au fil des années, le système d'exploitation a proliféré grandement dans le marché de la téléphonie [11].

Android a commencé son voyage avec le HTC. Bien que des caractéristiques pas si impressionnantes ne puissent pas donner la plate-forme Android nécessaire pour entrer sur le marché, il s'est montré assez prometteur par la suite.

Et c'est exactement ce qui s'est passé en 2011, lorsque le système d'exploitation Android mature et plus évoluée acquis près de la moitié de la part du marché (d'après le rapport de Technorati). Le rapport a également indiqué qu'Android ne pouvait pas faire détrôner les ventes d'iPhone d'Apple, mais Windows Phone et BlackBerry sont devenus des ombres dans le marché des appareils mobiles. Mais cela n'a pas duré longtemps, Android OS a vite pris de l'ampleur ces dernières années pour finir au sommet [12].

Nous illustrons son évolution sous différents angles à l'aide des figures suivantes :

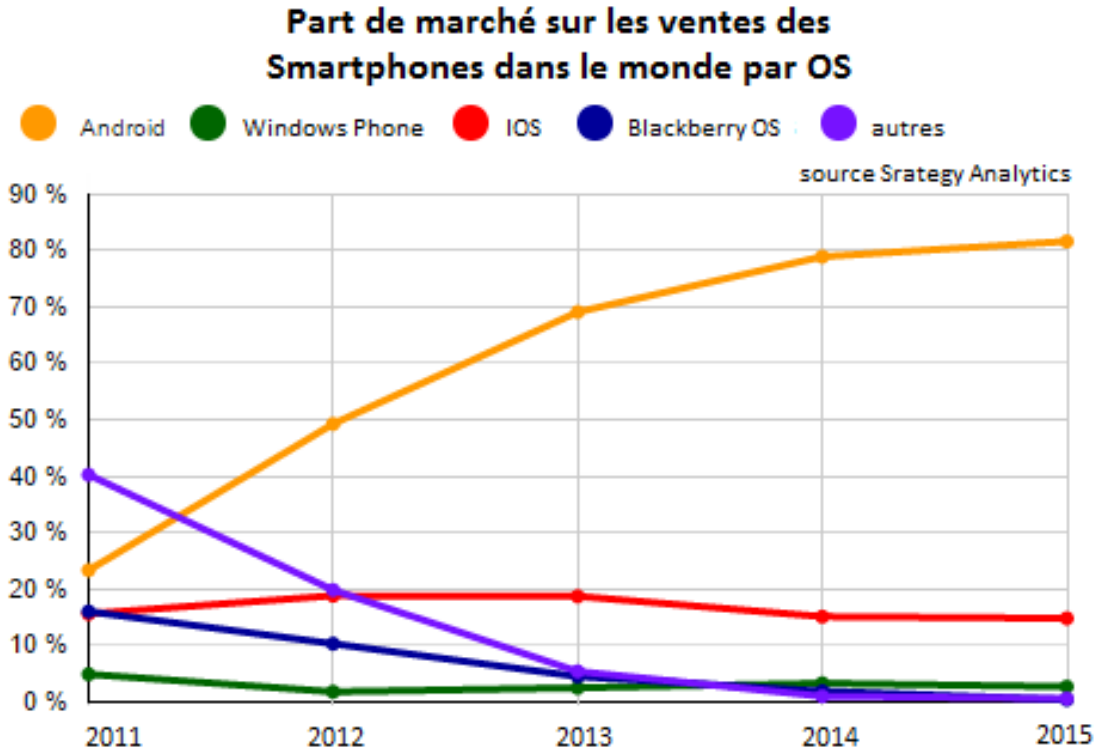


Fig. II.3 - Evolution du part de marché de l'Android par vente des Smartphones

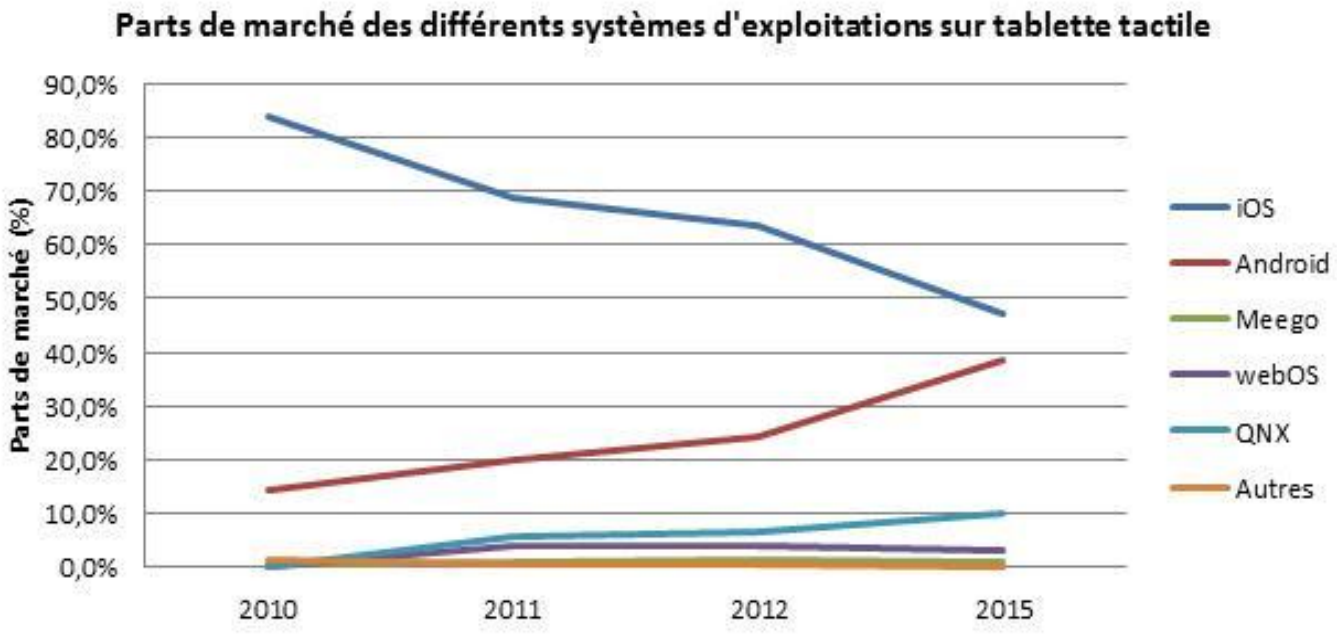


Fig II.4 - Evolution du part de marché par ventes de tablettes

5. Android vs les autres OS: Vision comparative

Ces systèmes d'exploitation peuvent être installés sur de nombreux modèles de téléphones différents, exceptionnellement pour l'iOS qui peut être intégré uniquement aux terminaux de la marque Apple, et généralement chaque appareil peut recevoir de multiples mises à jour du système d'exploitation durant sa durée de vie. Android de Google et iOS d'Apple fournissent non seulement le système d'exploitation, mais aussi une plate-forme de développement mobile parce que les deux sont confrontés à une rude concurrence. Maintenant, nous allons parler des systèmes d'exploitation principaux, respectivement Android, iOS et Windows Phone qui sont fréquemment observés sur le marché, en faisant une comparaison et on abordera leurs points forts et faibles ainsi que d'autres aspects.

5.1Android :

Les points forts :

- Communauté : Le point fort d'un système open source, c'est qu'il bénéficie très vite, d'une communauté de passionnés souhaitant apporter leur petite pierre à l'énorme édifice Android.
- Ergonomie et prise en main : La prise en main de l'OS peut être compliquée pour un grand néophyte qui n'est pas habitué aux terminaux tactiles.
- Market : La grande force de l'Android c'est son Appstore. On y trouve beaucoup d'applications et la pluparts sont gratuites.

Les points faibles :

- Stabilité : Le point fort d'Android, qui est également un point faible, c'est l'open source. Si l'on rentre trop dans la « bidouille » du système, on peut vite obtenir un système très instable (fermeture inopiné d'applications, reboot du système ou même certaines fonctionnalités inaccessibles).
- Ergonomie et prise en main : Comme énoncé plus haut, le point faible d'Android sera son interface qui peut dérouter les néophytes.
- Market : Sur le Playstore, on trouve tout et n'importe quoi (des applications sans grandes utilités).

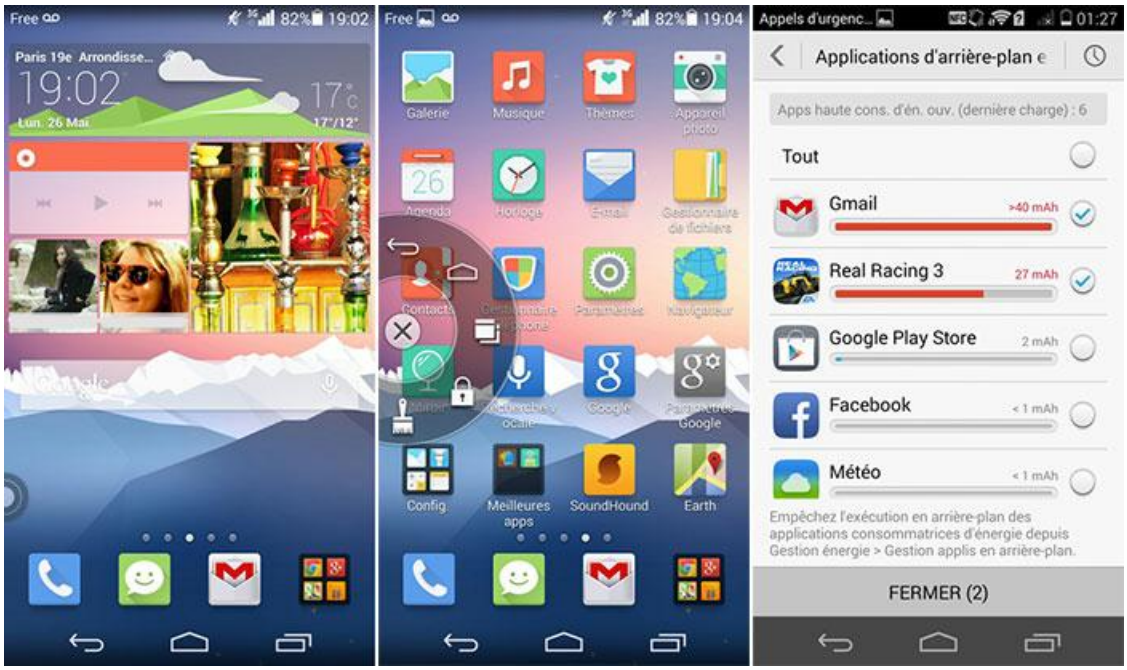


Figure II.5 - L'interface utilisateur du Huawei Ascend P7.

5.2 iPhone OS (iOS) :

Apple a souhaité rester maître du matériel comme du logiciel, ce qui signifie qu'ils vendent un téléphone unique équipé d'un système d'exploitation unique. Cela leur permet de maîtriser l'un comme l'autre et de pouvoir prendre bien plus de décisions par rapport au système.

Les points forts :

- Stabilité : Le système a fait ses preuves depuis la sortie de la première version en juin 2007. Si l'on imagine une utilisation de base d'iOS, le système est stable à 99%. La stabilité du système vient aussi de l'écosystème Apple, qui permet un contrôle quasi-total de bout à bout (de la création à la vente).

- Ergonomie et prise en main : En étant le plus objectif possible, un néophyte sera à l'aise dans l'interface (utilisation et paramétrage de base), mais seulement après quelques explications. Les utilisateurs familiers des interfaces tactiles, se débrouilleront très rapidement et facilement.
- Store : Apple possède à l'heure actuelle, l'un des plus grand catalogue de musique et d'applications mobiles. En ce qui concerne les livres et le magasin d'applications Mac, le contenu se densifie de jour en jour.

Les points faibles :

- Système : L'un des plus grands points forts d'iOS est également son plus gros point faible (au niveau du système). L'écosystème Apple rend hermétique iOS. Il est jugé beaucoup trop fermé par les utilisateurs, et pour permettre des fonctionnalités supplémentaires, il faut en passer par le Jailbreak. Or, ceci entraine des instabilités et des failles de sécurité.
- Interface : l'interface graphique n'a pratiquement pas évoluée depuis des versions. Seulement quelques modifications d'icônes, l'ajout du centre de notifications, et quelques petits effets d'animations, ont permis une relative petite évolution d'iOS. Les ajouts que la concurrence incorpore à son système sont toujours absents d'iOS (tels que les widgets, les gestures avancés, etc).
- Autonomie : Apple étant son propre fabricant de terminaux, il devrait y avoir une certaine synergie entre iOS et le matériel. On peut dire que c'est globalement le cas, sauf au niveau de la batterie. Et il n'est pas surprenant de voir Apple sortir une mise à jour mineure suite à sa mise à jour majeure, car beaucoup d'utilisateurs se plaignent de soucis d'autonomie.

Fig. II.6 L'interface utilisateur d'iPhone 5

Windows Mobile :

Les points forts :

- Stabilité : on peut parler d'un système « relativement » stable. A cause du nombre multiple de constructeurs, la stabilité est variable selon les terminaux.
- Ergonomie et prise en main : L'ergonomie reste simple d'utilisation avec son écran principal et l'écran listant l'intégralité des applications du terminal.
- Evolution : Malgré la jeunesse de l'OS, Microsoft a su faire évoluer son système pour correspondre au mieux aux attentes utilisateurs
- Market : le Windows Store est le dernier arrivé des market. Outre les classiques qui ont cartonné sur l'AppStore ou sur le Playstore, on va retrouver quelques exclusivités à Microsoft.

Les points faibles

- Ecosystème Microsoft : À l'image d'Apple, un système trop fermé, représente le plus gros des points faibles.
- Ergonomie et prise en main : Si l'écran d'accueil part d'un bon esprit, l'écran secondaire de l'interface est nettement moins accueillant. Il se contente de lister tout le contenu du terminal.
- Market : Les prix et le contenu du Windows Store mobile sont très discutables.

Microsoft souffre de la jeunesse de son OS, mais également d'un retard sur la concurrence qui sera très compliquée à remonter [13].

Fig. II.7 L'interface utilisateur du Nokia Lumia 1520.

6. Architecture

Android repose sur diverses technologies open source pour fournir une stabilité à sa plate-forme qui peut satisfaire les besoins mobiles. L'architecture de la plate-forme peut être décrite comme une série de cinq couches principales qui traitent des opérations différentes. La Figure suivante montre l'architecture avec ses niveaux et ses sous-niveaux :

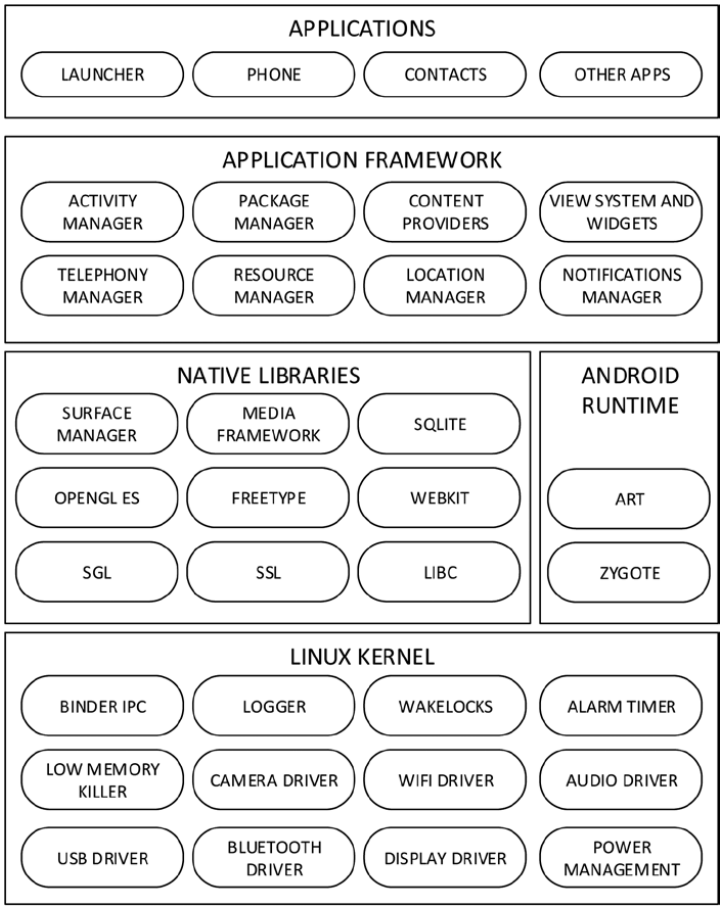


Fig.II.8 - Architecture du système Android

II. La révolution du code QR

1. Introduction

Le code-barres est la représentation d'une donnée numérique ou alphanumérique sous forme d'un symbole constitué de barres et d'espaces dont l'épaisseur varie en fonction de la symbologie utilisée (système de transposition entre un texte et un code-barres par un codage) et des données codées pour faciliter l'accès à l'information. On distingue les codes-barres unidimensionnels (1D) traditionnels qui sont représentés par des barres verticales, la demande accrue en plus d'information a guidé leur évolution en codes-barres bidimensionnelle (2D).

Codes-barres unidimensionnels (1D) [EAN, CUP, Code 11, Code 39…]	Codes-barres bidimensionnels (2D) [QR-Code, Data Matrix, FlashCode…]
	
Nombre limité d'informations encodées	Nombre important d'informations encodées : 25 à 100 fois plus
Dimensions limitées	Dimensions illimitées
Bit de contrôle / checksum	CRC 16/32
Pas de correction d'erreur	Différents niveaux de redondance

Fig. II.9 : Comparatif entre les générations de code-barres

2. Présentation du code QR

Le code QR est l'un des codes-barres à deux dimensions qui signifie qu'il est balayé dans les deux directions verticale et horizontale. Il fut créé et protégé par la société japonaise Denso Wave en 1994, et plus tard en an 2000 reconnu et défini par la norme industrielle ISO / IEC 18004. QR est l'abréviation de *Quick Response* en anglais, soit réponse rapide, qui met l'accent sur sa capacité à lire et décoder à grande vitesse d'où son autre appellation : le *Flash Code* [14].

3. Caractéristiques du code QR

- **Capacité** :

A la différence d'un code-barres classique, le code QR peut contenir beaucoup d'informations: 7089 caractères numériques, 4296 caractères alphanumériques, contrairement au code-barres normal qui ne peut stocker que de 10 à 13 caractères ou 2953 octets.

- **Structure du Code** :

Comme présenté précédemment le code QR étant bidimensionnel il consiste d'un rangement de carrés noir est blanc.

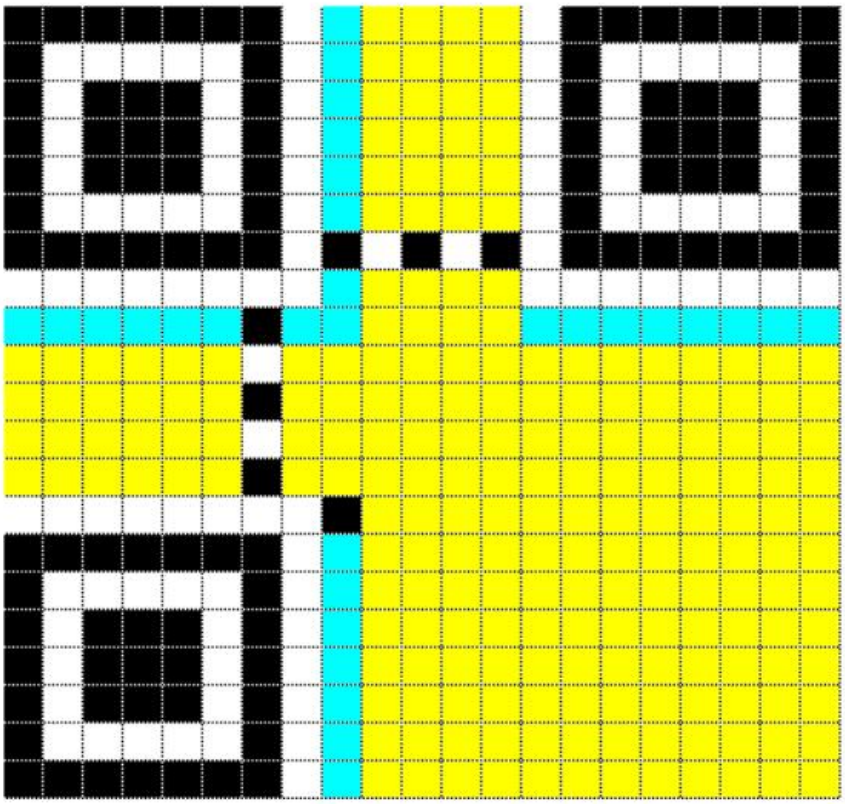


Fig. II.10 : Structure du code QR

La résolution du code, ou le nombre de carrés par rangée, peut différer pour contenir plus d'information pour une plus grande résolution. Cependant, il faut prendre en compte la sensibilité du capteur pour ne pas en résulter une mitigation d'information ou plus encore un code illisible.

- **Comparaison avec les autres codes bidimensionnels**

Le code QR fut populaire qu'en Japon à ses débuts avec son standard (JIS X 0510) mais dès sa normalisation, il s'est vu se répandre mondialement très rapidement pour devenir le modèle le plus utilisé. Ci-dessous un tableau illustrant les codes bidimensionnels les plus employés.

		QR Code	PDF417	DataMatrix	Maxi Code
					
Developer(country)		DENSO(Japan)	Symbol Technologies (USA)	RVSI Acuity CiMatrix (USA)	UPS (USA)
Type		Matrix	Stacked Bar Code	Matrix	Matrix
Data capacity	Numeric	7,089	2,710	3,116	138
	Alphanumeric	4,296	1,850	2,355	93
	Binary	2,953	1,018	1,556	
	Kanji	1,817	554	778	

Fig. II.11 : Les différents codes-barres bidimensionnels [15]

- **Lecture du Code**

En vue de sa complexité par rapport au code-barres classique, le code QR nécessite un capteur plus sophistiqué (*Fig. II.12)*.

Fig. II.12 Scanner QR sans fil à laser

Néanmoins avec la révolution des Smartphones, sa lecture est devenue accessible à tout le public par un simple scan avec l'appareil photo associée et un module préalablement installé (*Fig II.13*).

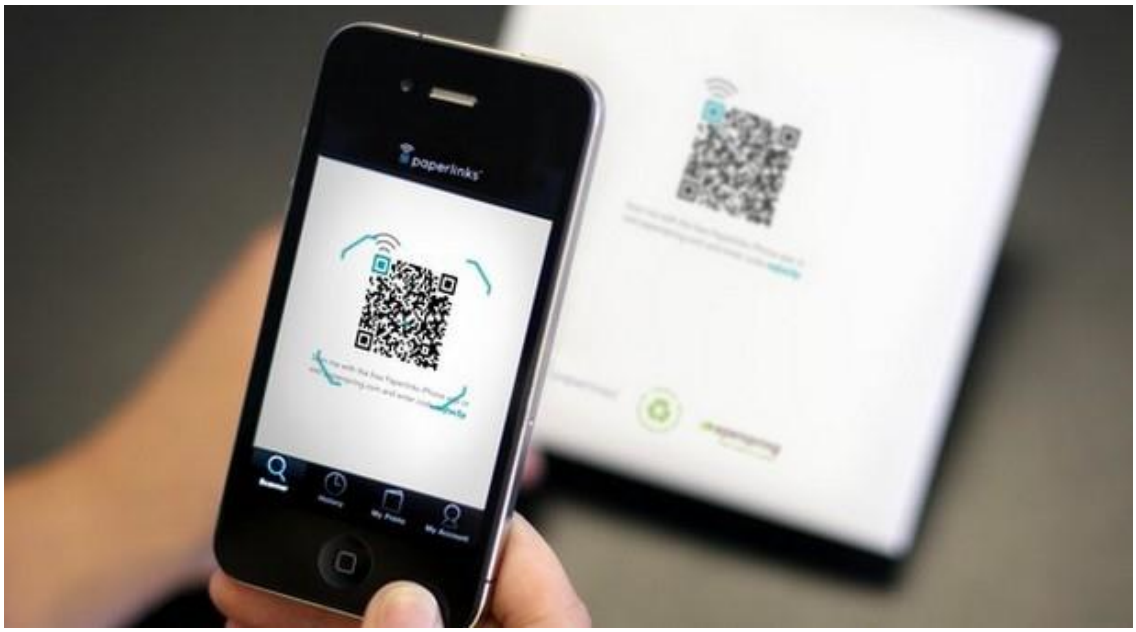


Fig. II.13 Scan d'un code QR avec *paperlinks* sur iPhone

Les modules de lecture sont compatibles avec n'importe quel système d'exploitation smartphone (Android, iOS..) tant que le terminal y est compatible.

- **Codage**

Pour coder une information en code QR il existe plusieurs générateurs, la plupart étant en ligne. On illustre un exemple : **« GOQR.me »** dans la figure ci-dessous.

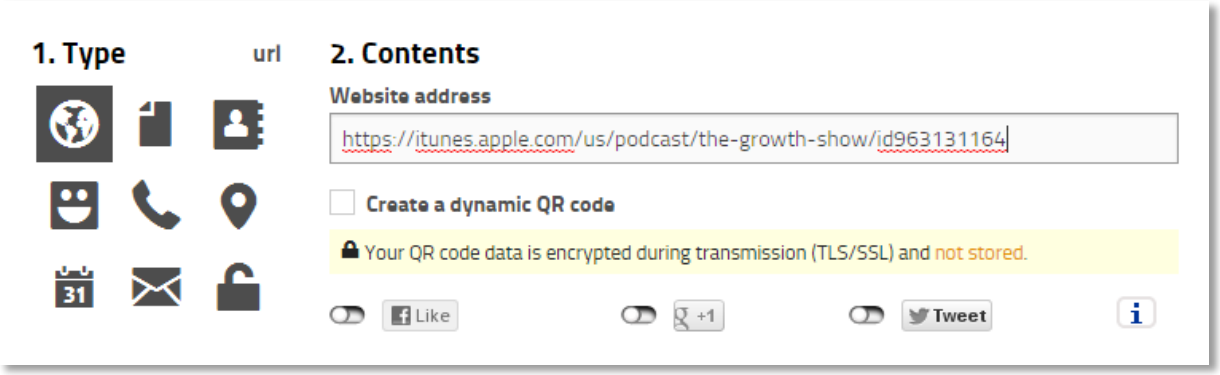


Fig. II.14 Génération d'un code QR en ligne

Il suffit de choisir le type (Lien site web, Informations de contacte, Emplacement GPS, Mail…), ensuite entrer l'information souhaitée selon le type et valider pour avoir le code QR qu'on peut customiser en y rajoutant son propre logo (*Fig. II.14*).

1

Fig. II.14 Aperçu du code QR généré

4. Déploiement et utilisations du code QR :

Il sert à coder une adresse URL comme par exemple celle d'un blog, en les lisant, on peut accéder très rapidement, aussi à coder un produit, un SMS, Email à envoyer, contact (nom, prénom, adresse, numéro téléphonique d'une personne pour l'enregistrer rapidement dans la liste des contacts après avoir scanné le QR Code via la caméra du mobile), etc.

En vue de la prolifération des smartphones, le code QR s'est vu vaguement répandu dans différents domaines d'expertise, on note :

- En domaine de publicité en dirigeant le lecteur vers le site de leur produit.
- Comme méthode de payement, exclusivement au Japon.
- Opérations d'enregistrements : Visas, tickets de train et autres, ou encore ouvrir des portes.

Fig. II.15 : Code QR sur Visa et ticket de Train

- **Accès à un réseau** : le balayage d'un code permet de rejoindre un réseau ou établir un pont de connexion entre le smartphone et l'ordinateur. La figure incontournable de ce service est AirDroid qui permet un accès total et sans fil d'un smartphone à partir d'un ordinateur.

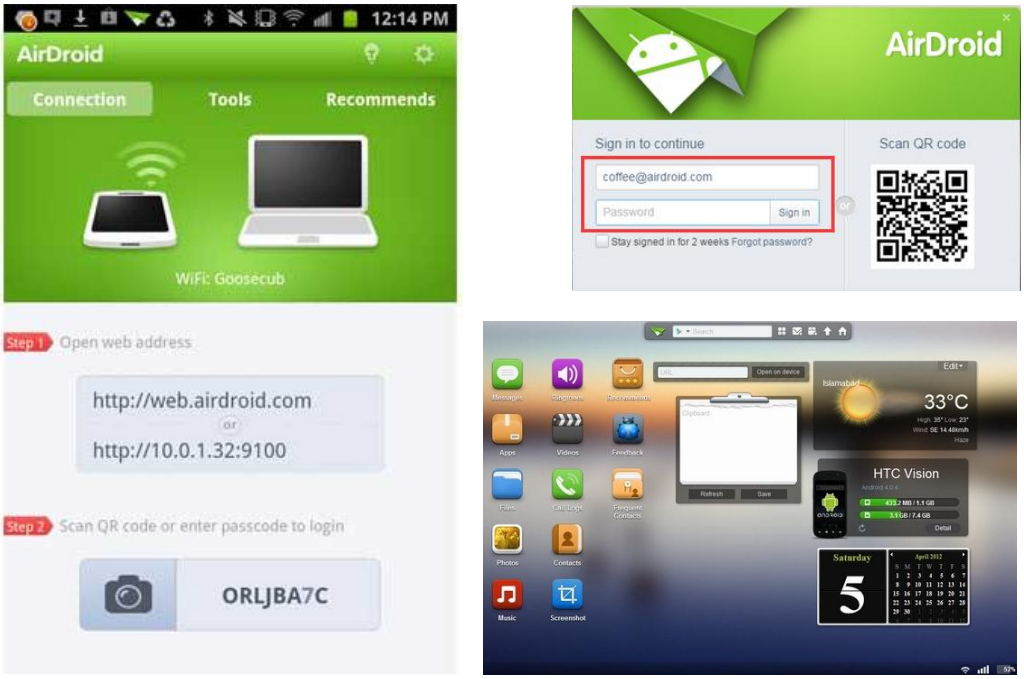


Fig. II.16 : Liaison d'accès par AirDroid

- Traçage de pièce que ça soit informatique, mécanique ou autres.
- Focalisation de notre étude : traçage des médicaments et substances, contenant les diverses informations associées au produit.

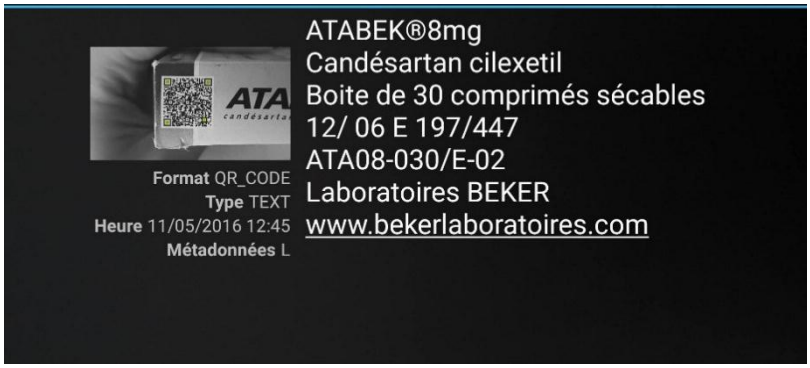


Fig. II.17 Scan d'un code QR sur un médicament

Conclusion :

Ce chapitre a consisté en premier sur la présentation du système d'opération dominant « Android », ses traits les plus importants ainsi qu'un comparatif avec ses rivaux dans le marché. On a également introduit le révolutionnaire en domaine de code-barres, le code QR avec toutes ses caractéristiques et son grand apport dans la technologie d'information en ouvrant la porte pour l'optimisation d'innombrables services existants.

Introduction :

L'aboutissement au développement d'une application Android nécessite plusieurs outils qui sont mis à disposition du développeur pour but d'accomplir l'objectif. Sur ce, les outils impliqués sont sensiblement liés à la nature de l'application et des fonctionnalités qu'elle offre. Dans ce chapitre, on présentera les différents outils exploités dans le développement de notre application.

I. L'environnement de développement

Depuis qu'Android a connu un tel succès, sa propagation a attisé l'apparition de plusieurs environnements de développement. Son développement, étant exclusivement disponible par Eclipse à ses commencements, est maintenant pris en charge par plusieurs nouvelles plateformes dont on cite les plus illustres :

1. **Android Studio** :

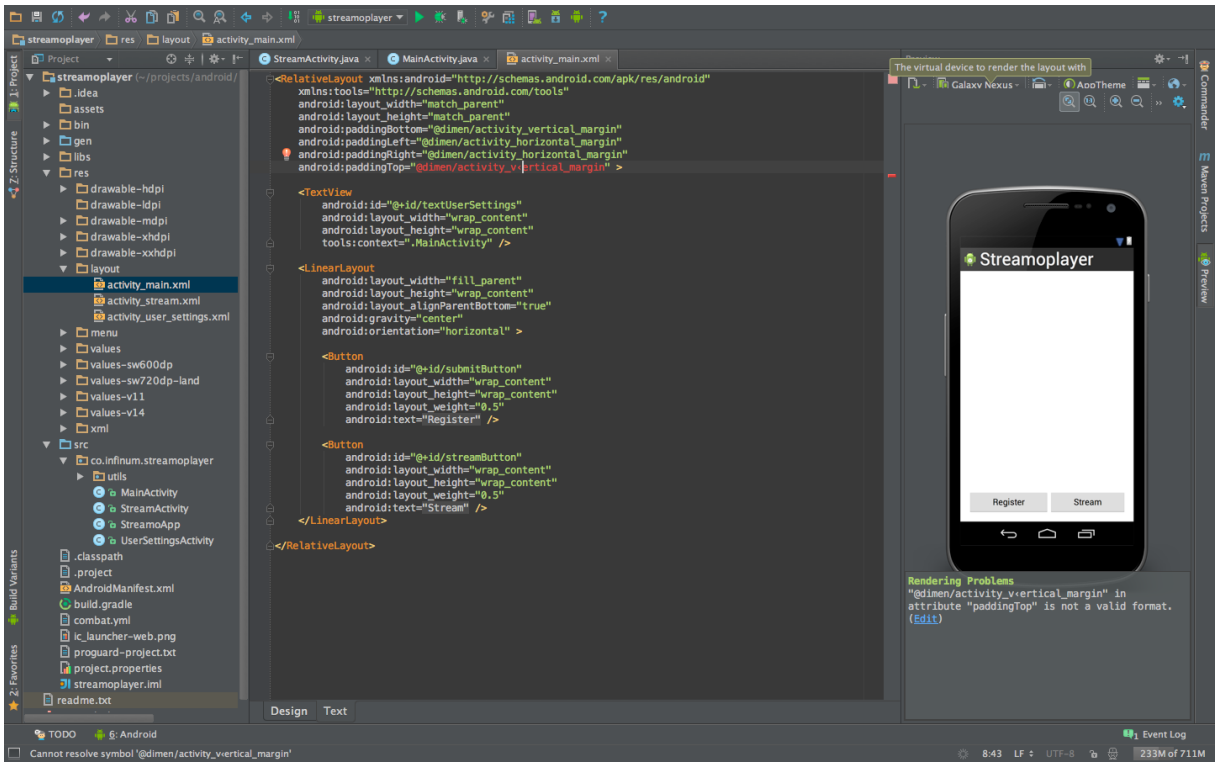


Fig. III.1 Interface d'Android Studio

Développé par Google, il est le plus populaire actuellement du fait qu'il soit entièrement dédié à l'Android évitant tout embrouillement au cours du développement. Son interface est sensiblement similaire à celle d'Eclipse. Il est également considéré être le plus rapide en matière de compilation et exécution du projet [16].

2. **Apache Cordova** :

Fig. III.2 Logo d'Apache Cordova

Anciennement appelé Apache Callback ou PhoneGap, Cordova est un framework (ensemble cohérent de composants logiciels structurels) de développement mobile open source. Son aspect distinctif est le développement des applications mobiles multiplateformes (Android, iOS, WMobile) à partir du même code basé sur HTML, HTML5, CSS et JavaScript. Cordova est encore frais et peu exploité mais son potentiel est prodigieux par sa fluidité de conception ainsi que la facilité de concevoir des interfaces graphiques amplement plus esthétiques. Néanmoins, une maitrise approfondie des langages soutenus par Cordova est nécessaire vu le manque de documentations sur ce dernier [17].

3. **Visual Studio :**

C'est une suite de logiciels de développement pour Windows multiplateformes conçue par Microsoft. Sa version récente intègre de nouvelles fonctionnalités qui simplifient le processus de développement d'applications mobiles, de la conception au déploiement. Il se base essentiellement sur le langage Visuel Basic, C++, C# et C. Considéré beaucoup moins flexible en conception d'applications mobiles par rapport aux autres environnements de développement mais il demeure un outil solide et puissant pour des applications avancées [18].

3. Eclipse :

Eclipse est un environnement de développement intégré (IDE) pour développer des applications utilisant le langage de programmation Java et d'autres langages de programmation tels que C / C ++, Python, PERL, Ruby, etc.

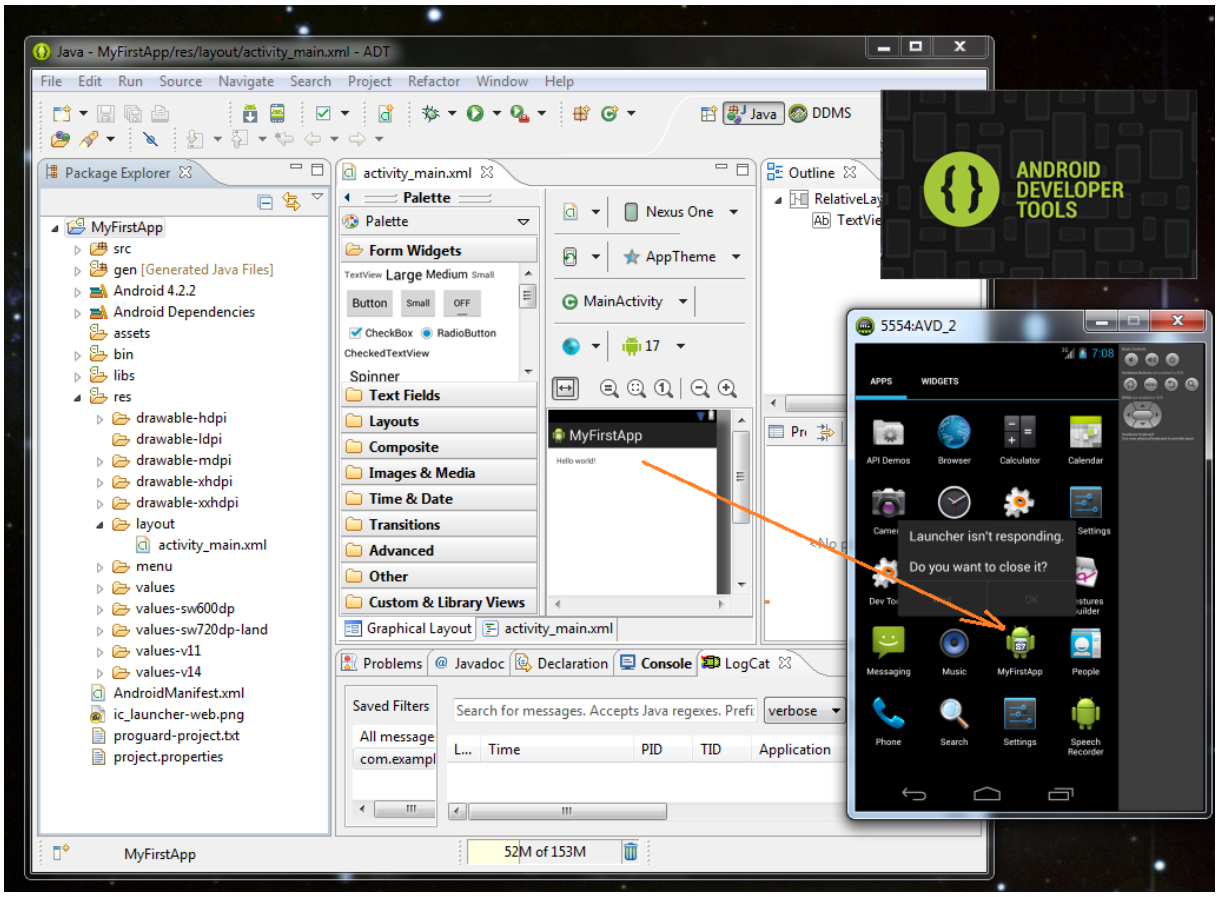


Fig. III.3 Interface Eclipse ADT avec l'émulateur

La plate-forme Eclipse qui constitue le fondement de l'IDE Eclipse est composée de plug-ins et est conçue pour être extensible à l'aide de plug-ins supplémentaires. Développé en Java, la plate-forme Eclipse peut être utilisée pour développer des applications riches, des environnements de développement intégrés et d'autres outils. Eclipse peut être utilisé comme un IDE pour tout langage de programmation pour lequel un plug-in est disponible [19].

L'aspect majeur d'Eclipse est sa flexibilité incomparable et son interopérabilité (la possibilité de connecter différents systèmes et langages). Avec la possibilité d'installer des centaines de plugins (petit logiciel qui se greffe à un programme principal pour lui conférer de nouvelles fonctionnalités), on peut adapter Eclipse à n'importe quel langage IDE disponible et sustenter

n'importe quel besoin nécessaire à l'aboutissement d'un projet d'où notre choix s'est porté sur cet IDE.

Le projet Outils de Développement Java (JDT) fournit un plug-in qui permet à Eclipse d'être utilisé comme un IDE Java. PyDev est un plugin qui le permet d'être utilisé comme un IDE Python, C / C ++ Development Tools (CDT) pour le développement de l'application en utilisant C / C ++.

Le plugin nécessaire dans notre application est l'ADT (Android Développement Kit) qui ajoute les fonctionnalités nécessaires à développer une application Android en intégrant le DDMS et les bibliothèques Android à travers l'Android SDK Manager (Android Software Développement Kit) ainsi que la gestion de la machine virtuelle Android à travers l'AVD Manager (Android Virtual Device).

En parlant de plugin, le M2Eclipse (Maven to Eclipse) se fait distinguer dans la version récente d'Eclipse (nommée Mars) en apportant des fonctionnalités qui permettent un développement plus fluide et une importation de bibliothèques externes plus autonome.

Maven :

Maven est un outil de construction de projets (*Build Management Tool*) open source développé par la fondation Apache. Il définit la façon dont les fichiers « .java » sont compilés en « .class » ou emballé en « .jar ». Il est similaire à *Apache Ant* ou G*radle*, *Makefiles* en C/C++, mais il tente d'être complètement autonome de telle manière de ne pas avoir des outils ou des scripts supplémentaires en incorporant d'autres tâches courantes comme le téléchargement et l'installation de bibliothèques nécessaires, etc [20].

Il est également conçu pour assurer l'aspect « portabilité de construction » de sorte qu'il n'y ait pas de problème de compatibilité du même code dans différents systèmes.

Dans notre projet, on a utilisé Maven pour intégrer le lecteur de code QR dans notre application afin qu'elle soit indépendante d'une application externe comme l'est le cas pour la plupart des applications Android exploitants cette fonctionnalité.

Architecture de base d'un projet

L'arborescence du projet est découpée en plusieurs éléments. Tous les fichiers sources se trouvent dans le répertoire *src*. Une fois le projet compilé, on devrait observer l'ajout d'un fichier *R.java*, qui vous fournira toutes les références des ressources incluses dans le répertoire *res*. À l'instar de tous les fichiers générés, le fichier *R.java* ne sera mis à jour que si l'intégralité de l'application se compile correctement.

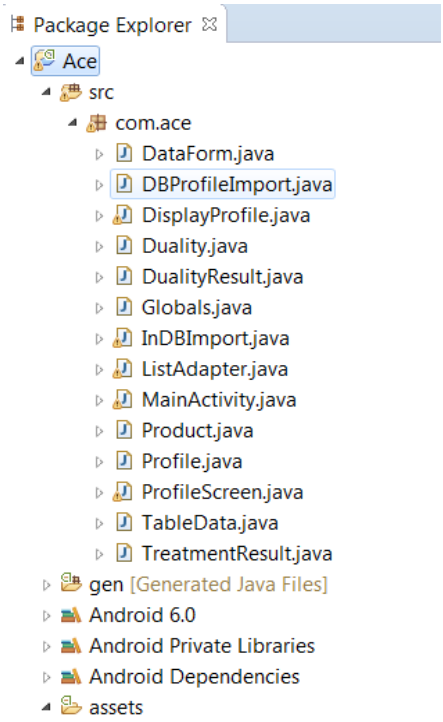


Fig. III. 4 - Architecture d'un projet

Src	Le répertoire de l'ensemble des sources du projet. C'est dans ce répertoire que vous allez ajouter et modifier le code source de l'application.
Libs	Contient les bibliothèques tierces qui serviront à votre application.
res	Contient toutes les ressources telles que les images, les dispositions de l'interface graphique, etc. nécessaires à l'application. Ces ressources seront accessibles grâce à la classe R décrite plus bas.
Gen	Contient l'ensemble des fichiers générés par ADT afin d'assister le développeur. Si vous supprimez un fichier dans ce répertoire, ADT s'empressera aussitôt de le régénérer.
Assets	Contient toutes les ressources brutes (raw bytes) ne nécessitant aucun traitement par ADT ou Android. À la différence des

	ressources placées dans le répertoire res, les ressources brutes seront accessibles grâce à un flux de données et non grâce à la classe R décrite plus loin.
Android Manifest.xml	Le fichier XML décrivant l'application et ses composants – activités, services, etc.
default. properties	Fichier de propriétés utilisé pour la compilation.

Tableau II.1 - Les éléments principaux dans un projet Android [21]

Le répertoire *res* est composé de différents sous-répertoires dont les suivants :

• *res/drawable* : contient les ressources de type image (PNG, JPEG et GIF) ;

• *res/layout* : contient les descriptions des interfaces utilisateur ;

• *res/values* : contient les chaînes de caractères, les dimensions, etc. ;

• *res/xml* : contient les fichiers XML supplémentaires (préférences, etc.) ;

• *res/menu* : contient la description des menus ;

• *res/raw* : contient les ressources autres que celles décrites ci-dessus qui seront empaquetées sans aucun traitement.

Layouts

Le composant graphique élémentaire de la plate-forme Android est la vue : tous les composants graphiques (boutons, images, cases à cocher, etc.) d'Android héritent de la classe *View*. L'utilisation et le positionnement des vues dans une activité se feront la plupart du temps en utilisant une mise en page qui sera composée par un ou plusieurs gabarits de vues. L'utilisateur peut décrire ses interfaces utilisateur soit par une déclaration XML, soit directement dans le code d'une activité en utilisant les classes adéquates. Dans les deux cas, ils peuvent utiliser différents types de gabarits. En fonction du type choisi, les vues et les gabarits seront disposés différemment :

• **LinearLayout** : permet d'aligner de gauche à droite ou de haut en bas les éléments qui y seront incorporés. En modifiant la propriété orientation vous pourrez signaler au gabarit dans quel sens afficher ses éléments : avec la valeur horizontale, l'affichage sera de gauche à droite alors que la valeur verticale affichera de haut en bas.

• **RelativeLayout** : ses éléments sont positionnées les uns par rapport aux autres, le premier enfant servant de référence aux autres ;

• **FrameLayout** : c'est le plus basique des gabarits. Chaque élément est positionné dans le coin en haut à gauche de l'écran et affiché par-dessus les enfants précédents, les cachant en

partie ou complètement. Ce gabarit est principalement utilisé pour l'affichage d'un élément (par exemple, un cadre dans lequel on veut charger des images) ;

• **TableLayout** : permet de positionner vos vues en lignes et colonnes à l'instar d'un tableau.

II. Langage

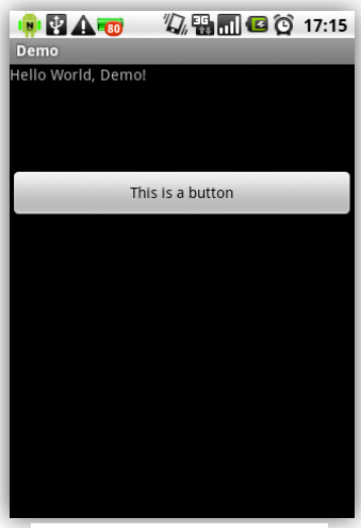
1. Basé sur Java

Les applications Android utilisent principalement la plate-forme Java pour s'exécuter. Il existe bien un framework natif nommé NDK (Native Development Kit) permettant de développer en C/C++, mais la difficulté de développer nativement des applications font que son utilisation restera certainement confidentielle, réservée à l'optimisation des performances des applications.

2. Interfaces et XML

Les interfaces utilisateurs sous Android sont définies dans un langage balisé de type XML1. Cette approche donne la possibilité au développeur d'écrire son interface, d'utiliser un outil de création d'interface ou encore de le développer lui-même. Ce format étant ouvert et humainement compréhensible, il est facilement traitable dans un script ou une application pour, par exemple, remplacer en masse un élément présent dans plusieurs interfaces.

```
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout
xmlns:android="http://schemas.android.com/apk/res/android"
      android:orientation="vertical"
      android:layout_width="fill_parent"
      android:layout_height="fill_parent"
      >
      <TextView
          android:layout_width="fill_parent"
          android:text="@string/hello"
android:layout_height="100px"/>
      <Button
            android:text="This is a button"
            android:id="@+id/Button01"
            android:layout_width="fill_parent"
            android:layout_height="wrap_content"
            />
</LinearLayout>
```



Sortie

Fig. III.5 : Exemple du code XML d'une interface utilisateur basique

La création d'une interface utilisateur sur Android se rapproche plus d'un code écrit en HTML qu'en C++. Bien qu'il demande une période d'apprentissage, il parait évident qu'elle est assez simple à prendre en main.

III. Les différents types d'applications sous Android

1. Activité

C'est l'application de base sous Android. Comme une application avec une interface pour l'utilisateur dépend toujours d'une activité, la plupart des applications en ont donc au moins une. L'activité va permettre d'afficher une vue contenants des contrôles. Ces contrôles sont simplement les listes, images, boutons, champs de saisie, etc. Dans le modèle MVC (Model, View, Controller), l'activité est le controller. C'est donc en quelque sorte le moteur de l'application. Une activité peut en ouvrir une autre, lui fournir des données et en récupérer à sa fermeture. Nous verrons plus en détail le fonctionnement des activités.

- **Cycle de vie d'une application Android**

Une activité n'a pas de contrôle direct sur son propre état (et par conséquent vous non plus en tant que programmeur), il s'agit plutôt d'un cycle rythmé par les interactions avec le système et d'autres applications. La figure suivante présente ce que l'on appelle le cycle de vie d'une activité, c'est-à-dire qu'il indique les étapes que va traverser notre activité pendant sa vie, de sa naissance à sa mort. Chaque étape du cycle est représentée par une méthode [22].

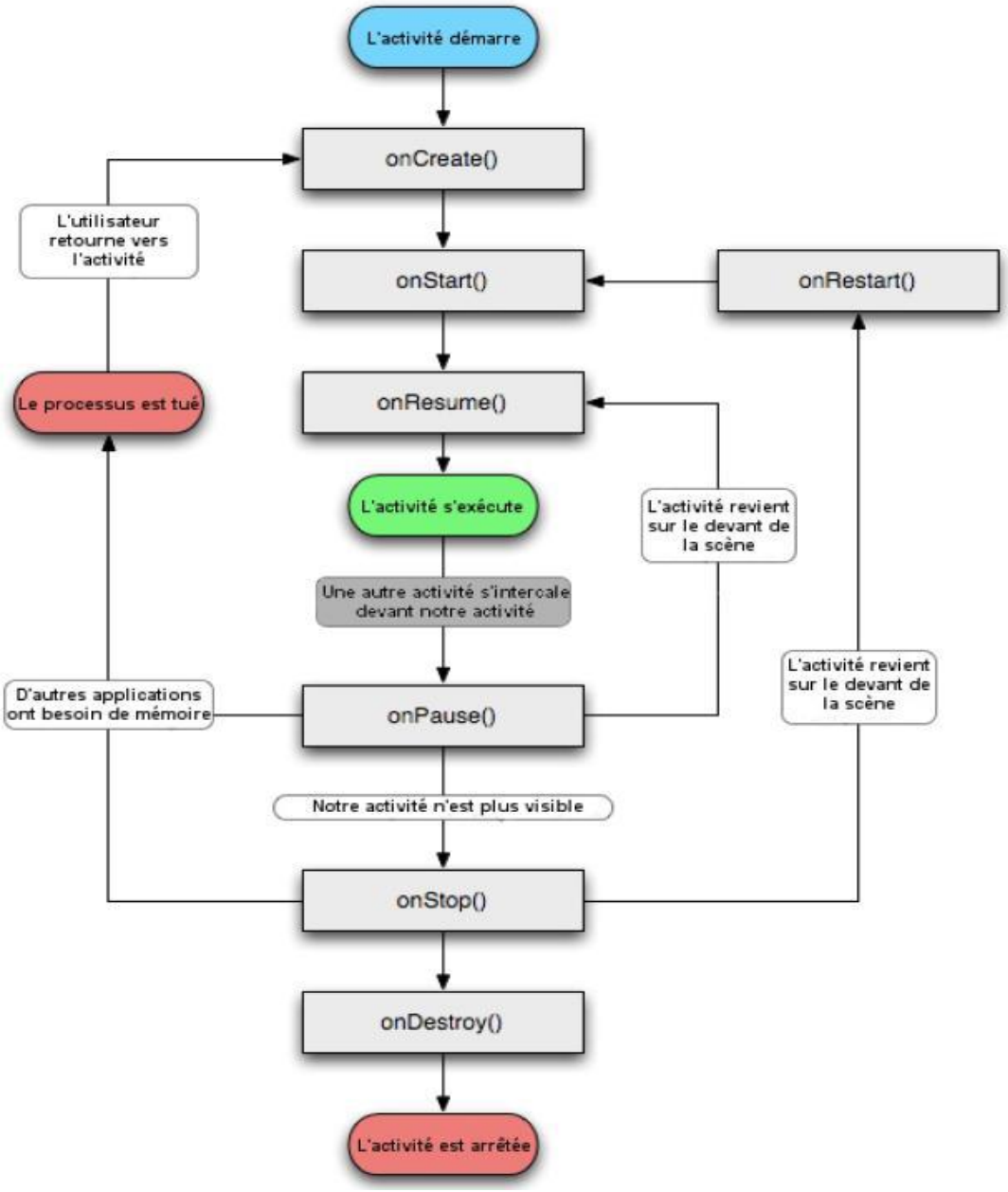


Fig. III.6- Cycle de vie d'une activité

2. Service

Un service est une application sans interface pour l'utilisateur puisqu'elle fonctionne en arrière-plan. C'est justement avec ces services que le multitâche d'Android a un sens. Le service permettra par exemple de lire un morceau de musique tout en faisant autre chose. Ou encore continuer le téléchargement d'un fichier malgré la fermeture de l'application.

3. BroadcastReceiver

Les BroadcastReceiver sont une alternative aux services. C'est une sorte de service endormi qui se réveille suite à une demande spécifiée. Ils ont été créés pour répondre au besoin d'économie d'énergie que demandent les appareils comme les smartphones. Il est possible de placer des BroadcastReceiver sur des alarmes pour réaliser des actions différées ou périodiques comme la mise à jour d'un widget météo. Ils permettent aussi d'écouter des messages du système ou d'autres applications comme la réception d'un SMS. Il est bien sûr possible d'envoyer ses propres messages broadcast.

4. ContentProvider

Les ContentProvider, comme leur nom l'indique, sont des gestionnaires de données. Ils permettent la mise à disposition de données filtrables aux autres applications. Ils sont utilisés pour récupérer la liste des SMS ou encore la liste des contacts. Il est aussi possible de créer ses propres ContentProvider afin de rendre disponibles les données de l'application en question. Par exemple, pour une application de carte de visite, il peut être judicieux de proposer un accès contrôlé aux données constituant les cartes de visite [23].

IV. Les bases de données SQLite

1. Présentation du SQLite

SQLite est une bibliothèque qui met en œuvre avec une autonomie, sans serveur, sans configuration, le moteur de base de données SQL transactionnel. Le code pour SQLite est dans le domaine public et est donc libre pour un usage à des fins commerciales ou privés. SQLite est actuellement dans plus de demandes que nous pouvons compter, dont plusieurs projets de grande envergure. SQLite est un moteur de base de données SQL embarqué. Contrairement à la plupart des autres bases de données SQL, SQLite n'a pas de processus serveur séparé. SQLite lit et écrit directement dans les fichiers de disques ordinaires. Une base de données SQL avec plusieurs tables, index, triggers et les vues, est contenue dans un fichier de disque unique. Le format de fichier de base de données est multi-plateforme - vous pouvez librement copier une base de données entre les systèmes 64 bits et 32 bits ou entre architectures big-endian et little-endian. Ces caractéristiques font de SQLite un choix populaire comme un format de fichier de l'application [24].

2. SQLiteDatabase Browser

SQLiteDatabase Browser se présente comme un outil visuel utilisé pour la création et la conception des bases de données et de modifier des fichiers compatibles avec SQLite. C'est un éditeur graphique léger pour les bases de données SQLite, construit sur Qt. L'objectif principal du projet est de permettre aux utilisateurs non techniques de créer, modifier et éditer des bases de données SQLite en utilisant un ensemble d'assistants et une interface de type tableur.

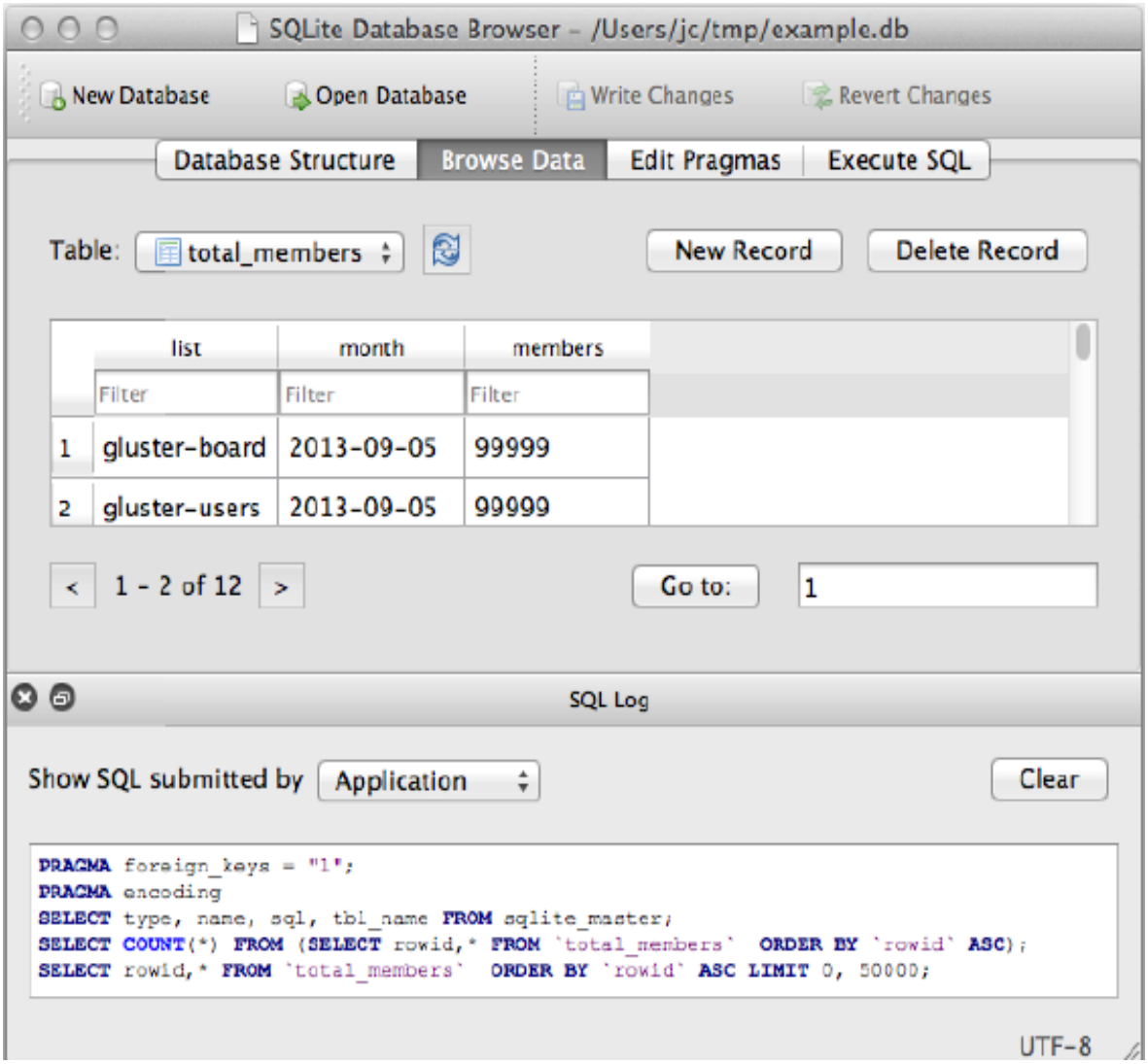


Fig. III.7- Interface du SQLiteDatabase Browser

Les contrôles et les assistants disponibles permettent aux utilisateurs de :

- Créer des fichiers de base de données compacts.
- Créer, définir, modifier et supprimer des tables.
- Créer, définir et supprimer les index.
- Parcourir, modifier, ajouter et supprimer des enregistrements.
- Recherche documents.
- Documents d'importation et d'exportation sous forme de texte.
- Importation et exportation de tables à partir de /vers des fichiers CSV.
- Importation et exportation de bases de données à partir de /vers des fichiers dump SQL.
- Émettre des requêtes SQL et inspecter les résultats.
- Examiner un journal de toutes les commandes SQL émises par l'application.

3. SQLite dans Android

Android permet l'utilisation de base de données SQLite6 et le SDK fournit tout le nécessaire pour créer et utiliser une ou plusieurs bases de données par application. Pour se faire, il faut simplement créer une classe héritant la classe « SQLiteOpenHelper » intégrée au « framework ».

Lorsque l'on étend la classe « SQLiteOpenHelper », il est indispensable d' « overrider » les méthodes « OnCreate » et « OnUpgrade » comme le montre l'exemple ci-dessous. La méthode « OnCreate » est exécutée à la première exécution de l'application et permet de créer les différentes tables constituant la base de données. La méthode « OnUpdate » est appelée pour mise à jour de la base de données suite à une mise à jour de l'application. Elle permettrait par exemple de réaliser des « ALTER TABLE ».

Conclusion

Ce chapitre s'est focalisé sur les différents outils utilisés pendant le développement de notre application que ça soit un outil essentiel comme l'environnement de développement, ou secondaire come le gestionnaire de base de données (ceci ne reflète pas la futilité de l'opération, mais juste qu'elle pourrait se faire sans l'aide d'un tel outil) mais sans citer des programmes facultatifs comme Adobe Illustrator (utilisé dans la conception des éléments du GUI) étant purement esthétique. Le choix diffère selon chaque développeur et ses critères, donc, on a choisi les programmes les plus adaptés à notre projet et surtout le rendant plus évolutif.

Introduction

Il existe différentes techniques et approches pour développer une application en général. On préfère considérer les conditions estimées être les plus difficiles en premier. Ainsi, on a une vision claire de savoir si c'est possible d'achever la partie prenante de notre application, sa réussite donnera un élan d'avance pour le développement. Les défis difficiles de notre application se focalisent autour de l'exploitation du lecteur de code QR (introduit au chapitre II) et les interactions avec différentes bases de données au sein de l'application. La complexité de leur intégration ainsi que le tas d'obstacles contournés ne nous permettent pas de mentionner toutes étapes et manœuvres du processus du développement. Cependant, on les présentera brièvement en trois parties :

1- Intégration du lecteur de code QR
2- Création, importation et exploitation des bases de données
3- Développement du GUI avec explication par différents scenarios.

I. Intégration du lecteur de code QR

Il existe différentes applications utilisant cette fonctionnalité. Toutefois, mise à part les applications dédiées à ce service, dépendent de ces dernières pour fonctionner. Pour élucider ça, l'application doit « appeler » l'application dédiée à lire des codes QR et nécessite qu'elle soit installée préalablement.

Dans notre application, on a opté pour qu'elle soit « StandAlone » c'est-à-dire qu'elle pourra lire et interpréter un code QR sans le besoin d'une autre application ou service déjà disponible sur l'appareil. Le réaliser ne fut pas aisé, dû aux innombrables contraintes de compatibilité des différents systèmes et APIs (*Android Package Index* : version de la bibliothèque d'Android).

Le module de lecture de code QR qu'on a choisi est le « Zebra Crossing » ou « ZXing Code Reader » à cause de la flexibilité de son code source, ses mis à jours et correctifs continus et sa viabilité du service tout en supportant l'ancien code-barres unidimensionnel.

Son intégration commence par importer son code source (nommé Core.Jar) par Maven Repository (voir Chapitre II), puis en créant un dossier d'application à part entière (qui peut être considéré comme une sous application) qu'on a nommé « CaptureActivity » qui contiendra les différents éléments du ZXing et s'occupera exclusivement d'assurer le service de lecture et interprétation du code QR.

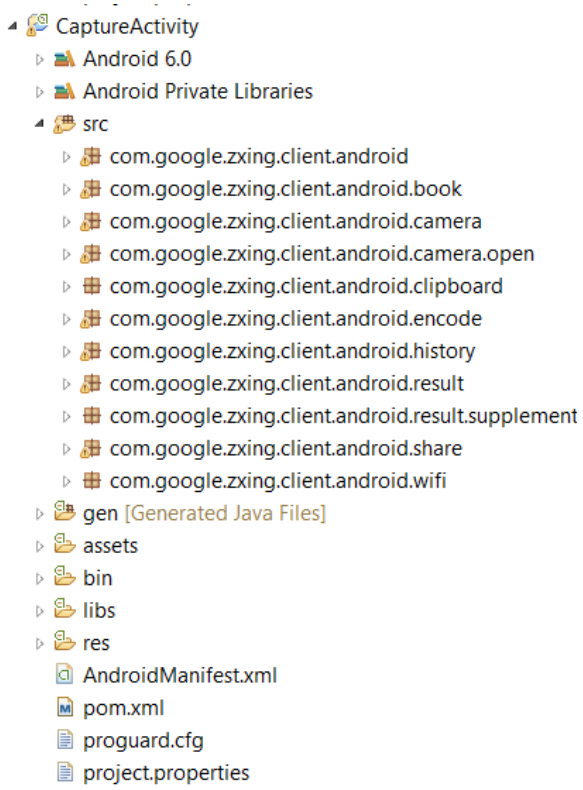


Fig. IV.1 : Arborescence de la partie du lecteur QR

Cette partie contiendra les classes Java et les Layouts responsables de la lecture et l'extraction des informations à partir du code QR en utilisant le « Core » du *ZXing Reader* importé par Maven précédemment.

Une fois l'implémentation réussie et le débogage effectué, le dossier « CaptureActivity » est définit en tant que bibliothèque pour pouvoir être appelée par la partie principale de l'application.

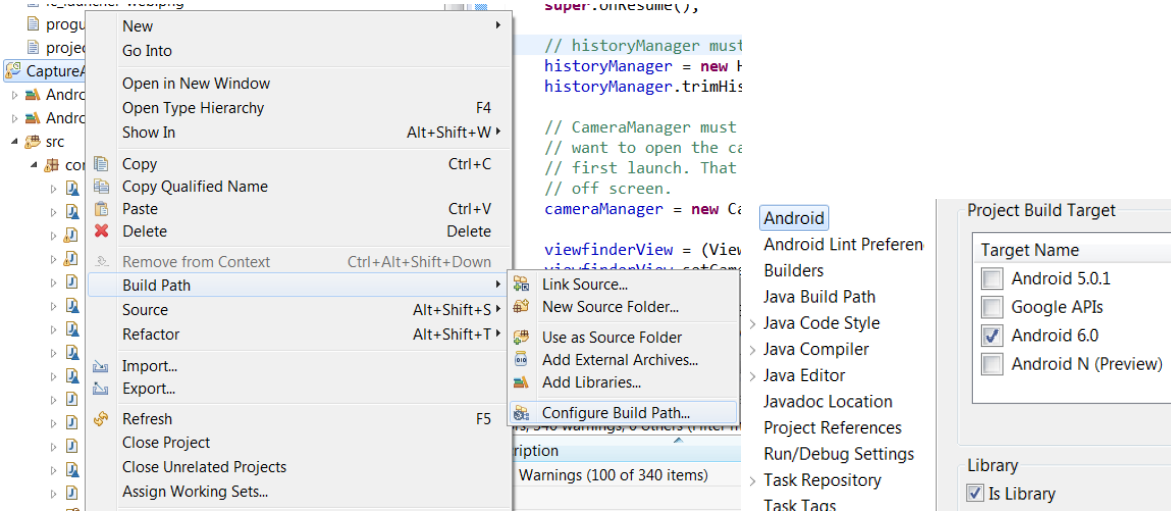


Fig. IV.2 : Définir CaptureActivity en tant que bibliothèque

L'ouverture du lecteur code QR se fait d'abord par importation du « CaputreActivity » par la commande suivante :

```
import com.google.zxing.client.android.CaptureActivity;
```

Ensuite, on crée un « Intent » dans la class « MainActivity » à l'intérieur du « KeyListener » du bouton correspondant :

```
// Lancement du lecteur QR Code
btCode = (Button) findViewById(R.id.btCode);
btCode.setOnClickListener(new View.OnClickListener() {

    @Override
    public void onClick(View v) {

    Intent intent = new Intent(getApplicationContext(),CaptureActivity.class);
    intent.setAction("com.google.zxing.client.android.SCAN");
    intent.putExtra("SAVE_HISTORY", false);
    startActivityForResult(intent, 0);
    }
});
```

Le résultat en cliquant le bouton associé au service va ouvrir l'appareil photo et afficher le rectangle pour cadrer le code-barres (Fig. IV.3)

Fig. IV.3 Ouverture du lecteur

Une fois le code-barres détecté, il sera scanné et décodé par les classes dans « CaptureActivity » pour être ensuite interprété grâce à la méthode *onActivityResult() :*

```
public void onActivityResult(int requestCode, int resultCode, Intent intent){
     if(requestCode == 0){
     if(resultCode == RESULT_OK){
     String contents = intent.getStringExtra("SCAN_RESULT");
     String format = intent.getStringExtra("SCAN_RESULT_FORMAT");
     Log.i("xZing", "contents: "+contents+" format: "+format); // Handle successful scan
     }
     else if(resultCode == RESULT_CANCELED){ // Handle cancel
     Log.i("xZing", "Cancelled");
     }
     }
    }
```

Le résultat effectué sera affiché ensuite si l'opération est réussite, on présentera l'exemple du médicament « ATABEK » (Fig. IV.4)

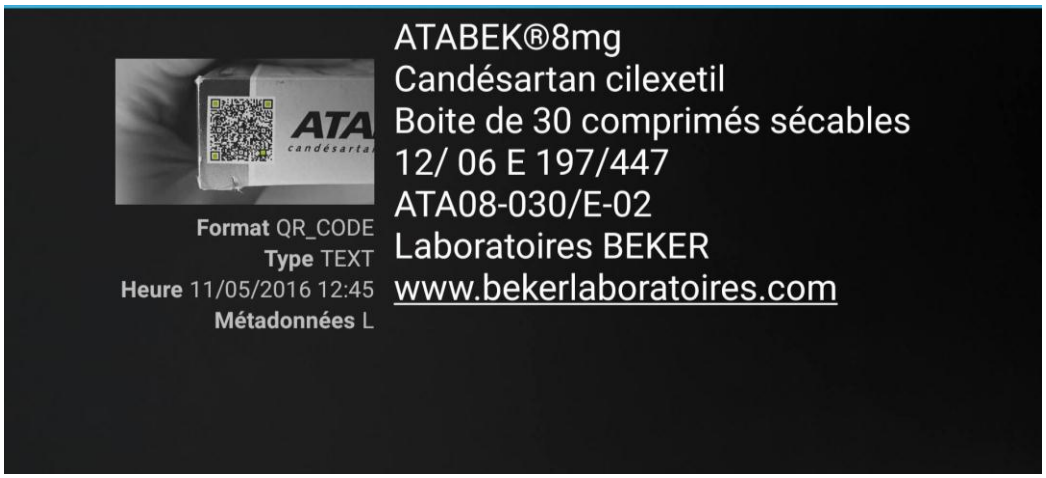


Fig. IV.4 Résultat du scan du médicament ATABEK

Dans le cas d'un code-barres unidimensionnel, le résultat contient un simple numéro de série montrant ainsi le contraste d'utilité et d'informations contenus entre les deux types de code-barres (Fig. IV.5).

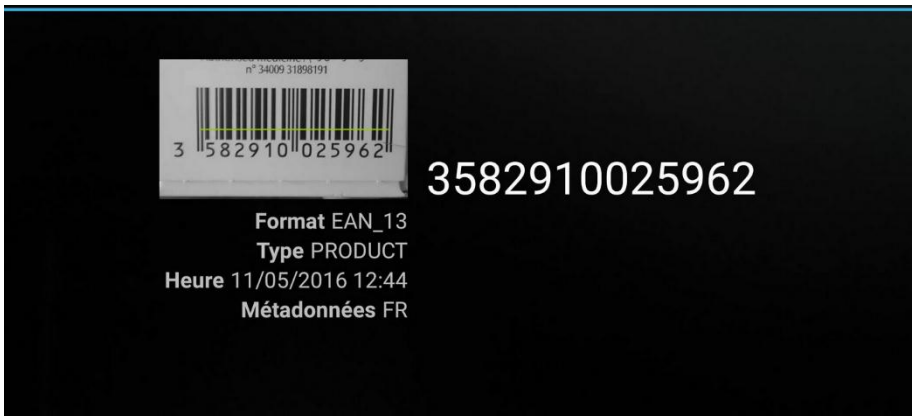


Fig. IV.5 Scan d'un médicament à code-barres unidimensionnel

Il faut noter que l'interprétation du code QR nécessite souvent une connexion internet pour pouvoir en retirer des informations supplémentaires et afficher le résultat en complet.

II. Création, importation et exploitation des bases de données

Le type de base de données utilisé est SQLite. Ce choix est évident vu la facilité de sa manipulation dans le système Android et sa rapidité du traitement.

On nommera deux bases de données distinctes :

***PDB* : Base de données des profils de traitement**

Elle contiendra les enregistrements des différents traitements entrés par l'utilisateur pour en faciliter l'exploitation de ces données (qui ne sont d'autre que la liste des médicaments pris en un seul traitement).

La base de données est créée dans le « SQLite Database Browser » et contient 3 tables :

Profile : contient les informations principales (id, Nom, liste des médicaments).

dbVersion : la version de la base de données nécessaires pour la mise à jour.

sqlite_sequence : table générée automatiquement pour tracer les lignes.

Nom	Type	Schéma
Tables (3)		
Profile		CREATE TABLE `Profile` (`id` INTEGER PRIMARY KEY AUTOINCREMENT, `NAME` TEXT, `MED` TEXT, `MED1` TEXT, `MED2` TEXT, `MED3` TEXT, `MED4` TEXT)
id	INTEGER	`id` INTEGER PRIMARY KEY AUTOINCREMENT
NAME	TEXT	`NAME` TEXT
MED	TEXT	`MED` TEXT
MED1	TEXT	`MED1` TEXT
MED2	TEXT	`MED2` TEXT
MED3	TEXT	`MED3` TEXT
MED4	TEXT	`MED4` TEXT
dbVersion		CREATE TABLE `dbVersion` (`version_id` INTEGER NOT NULL, PRIMARY KEY(version_id))
version_id	INTEGER	`version_id` INTEGER NOT NULL
sqlite_sequence		CREATE TABLE sqlite_sequence(name,seq)
name	TEXT	`name` TEXT
seq	TEXT	`seq` TEXT
Index (0)		
Vues (0)		
Déclencheurs (0)		

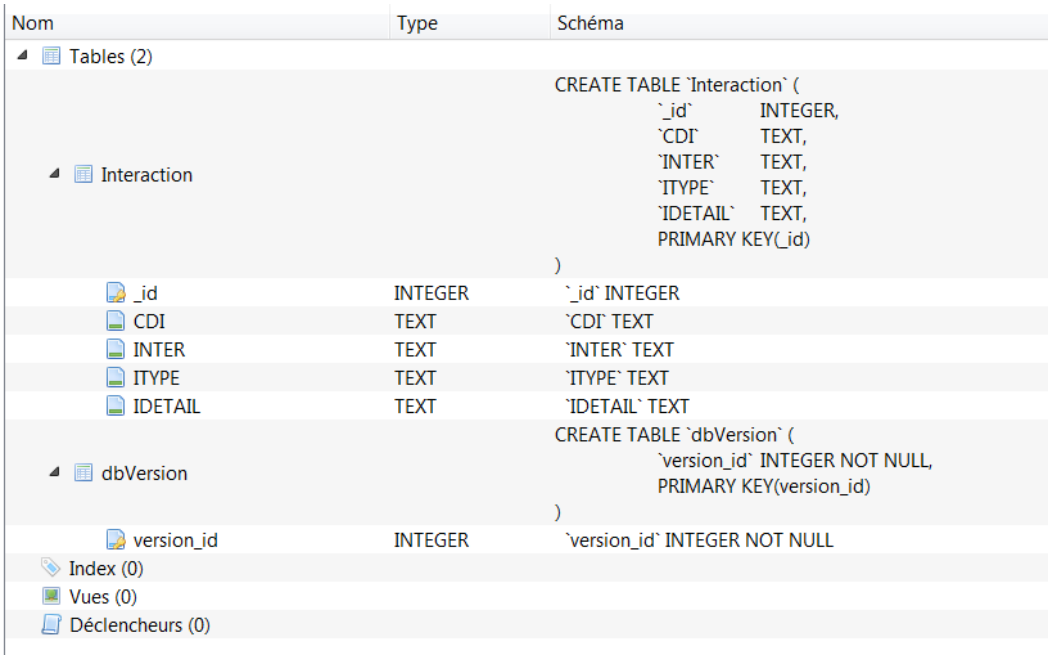
Fig. IV.6 : Création du PDB

Bien qu'on ait pu se contenter d'enregistrer les informations dans des champs et éviter la base de données, notre méthode a plusieurs avantages :

- Grandement optimiser les ressources du fait que les informations sont stockées dans un fichier est non dans la mémoire RAM.
- La possibilité d'avoir différents traitements enregistrés à la fois et les afficher dans une liste ergonomique.
- La rapidité d'accès aux informations.

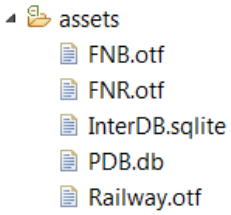
***InterDB* : Base de données des interactions**

C'est la partie la plus sensible dans notre projet. Il n'existe jusqu'à présent aucune base de données contenant toutes les interactions médicamenteuses de chaque nom commercial, et vu la complexité et le grand nombre d'interactions présentes, il est difficile d'automatiser tout le processus. C'est pourquoi on a commencé d'abord par les interactions *Drug-to-Drug* (un médicament avec un autre) en se basant sur « Thesaurus des interactions médicamenteuses » faites par l'ANSM (Agence nationale de sécurité du médicament et des produits de santé) avec la version de « Janvier 2016 » pour établir notre propre base de données nommée « InterDB.sqlite ». Dans un premier plan, cela permettra de comparer deux médicaments par leur CDI (Dénomination Commune Internationale ou nom de la molécule mère).



Nom	Type	Schéma
Tables (2)		
Interaction		CREATE TABLE `Interaction` (`_id` INTEGER, `CDI` TEXT, `INTER` TEXT, `ITYPE` TEXT, `IDETAIL` TEXT, PRIMARY KEY(_id))
_id	INTEGER	`_id` INTEGER
CDI	TEXT	`CDI` TEXT
INTER	TEXT	`INTER` TEXT
ITYPE	TEXT	`ITYPE` TEXT
IDETAIL	TEXT	`IDETAIL` TEXT
dbVersion		CREATE TABLE `dbVersion` (`version_id` INTEGER NOT NULL, PRIMARY KEY(version_id))
version_id	INTEGER	`version_id` INTEGER NOT NULL
Index (0)		
Vues (0)		
Déclencheurs (0)		

Fig. IV.7 Architecture primaire de l'InterDB



Ces bases de données seront dans des fichiers qu'on doit importer dans notre application qui seront compilé dans le fichier « APK » (Android Package Kit, le fichier d'installation de l'application à la sortie) qui sera ensuite copié vers le mobile Android. Les fichiers sont d'abord déposés dans le dossier « Assets » pour en avoir l'accès dans les classes Java.

L'importation de ces bases de données se fera avec l'intermédiaire d'une classe, connu sous le nom de « DatabaseHelper », pour chaque base de données. Elle contient les différentes procédures d'importations et de vérifications ainsi que les méthodes d'accès aux données. On citera les méthodes majeures de ces classes :

Déclaration du constructeur :

```
public class DBProfileImport extends SQLiteOpenHelper {
	private final static String TAG = "DatabaseHelper";
	private final Context myContext;
	private static final String DATABASE_NAME = "PDB.db";
	private static final int DATABASE_VERSION = 1;
	private String pathToSaveDBFile;
	public DBProfileImport(Context context, String filePath) {
		super(context, DATABASE_NAME, null, DATABASE_VERSION);
		this.myContext = context;
		pathToSaveDBFile = new StringBuffer(filePath).append("/").append(DATABASE_NAME).toString();
	}
```

Préparation à l'importation :

```
public void prepareDatabase() throws IOException {
	boolean dbExist = checkDataBase();
	if(dbExist) {
		Log.d(TAG, "Database exists");
		int currentDBVersion = getVersionId();
		if (DATABASE_VERSION > currentDBVersion) {
		    Log.d(TAG, "Database version is newer");
		    deleteDb();
		    try {copyDataBase();
		} catch (IOException e) {
		  Log.e(TAG, e.getMessage());
		}  }  } else { try {copyDataBase();
	  } catch (IOException e) {
	Log.e(TAG, e.getMessage());
    }  }  }
```

Méthode pour copier la base de données :

```
private void copyDataBase() throws IOException {
	OutputStream os = new FileOutputStream(pathToSaveDBFile);
	InputStream is = myContext.getAssets().open(DATABASE_NAME);
	byte[] buffer = new byte[1024];
	int length;
	while ((length = is.read(buffer)) > 0) {
		os.write(buffer, 0, length);
	}
	is.close();
	os.flush();
	os.close();
}
```

Extraction des données sous une forme de liste :

```
public List<Profile> getListProfile() {
  SQLiteDatabase db = SQLiteDatabase.openDatabase(pathToSaveDBFile, null, SQLiteDatabase.OPEN_READONLY);
  String query = "SELECT id, NAME FROM Profile";
  Cursor cursor = db.rawQuery(query, null);
  List<Profile> list = new ArrayList<Profile>();
  while(cursor.moveToNext()) {
      Profile profilee = new Profile();
      profilee.setId(cursor.getInt(0));
      profilee.setName(cursor.getString(1));
      list.add(profilee);}
  db.close();
  return list;}
```

Insertions d'une nouvelle ligne :

```
public void insertData(DBProfileImport pDB,String name,String med,String med1,String med2,String med3,String med4){
  SQLiteDatabase SQ = SQLiteDatabase.openDatabase(pathToSaveDBFile, null, SQLiteDatabase.OPEN_READWRITE);
  ContentValues cv = new ContentValues();
  cv.put("NAME",name);  cv.put("MED",med);   cv.put("MED1",med1);     cv.put("MED2",med2);
  cv.put("MED3",med3);  cv.put("MED4",med4);
  SQ.insert("Profile", null, cv);
  Log.d("Database add", "data inserted");
  SQ.close();}
```

Création d'un curseur :

```
public Cursor targetData(DBProfileImport pDB){
  SQLiteDatabase db = SQLiteDatabase.openDatabase(pathToSaveDBFile, null, SQLiteDatabase.OPEN_READONLY);
  String query = "SELECT * FROM Profile";
  Cursor CR = db.rawQuery(query, null);
  return CR; }
```

Quelques méthodes ne sont pas mentionnées vu la longueur du code comme la vérification de la version de la base de donnée, la suppression, la mise à jour et la vérification.

Ces méthodes sont utilisées à chaque fois que l'application en a besoin dans différentes classes. Toutefois, certaines opérations ne sont effectuées qu'une seule fois, on résume le déroulement de l'implémentation de la base de données par ce qui suit :

- Pendant l'ouverture de l'application pour la première fois, pendant l'affichage de l'interface principale (MainActivity), on vérifie si la base de données existe auparavant dans le mobile (les bases de données sont enregistrées dans un dossier système d'un lien du type « ../data/data/com.*NomApplication*/databases ») en trouvant aucune. L'opération se poursuit par une création d'un tunnel et un buffer comme pour une connexion réseau sauf que l'adresse est virtuelle et renvoi au *DatabasePath* (Lien de dépôt précédemment cité) et se finira par copier le fichier depuis le dossier Assets vers le dossier mentionné.

Cette opération n'est effectuée que pendant la première exécution de l'application ou en cas d'apporter une nouvelle version de base de données puisque il vérifie aussi la version de la base de données dans la table db_Version à chaque lancement.

```
InDBImport mDBI = new InDBImport(this, getFilesDir().getAbsolutePath());
try {
     mDBI.prepareDatabase();
} catch (IOException e) {
    Log.e(TAG, e.getMessage());
}
```

- Une fois le fichier de bases de données copié, l'accès sera à travers les méthodes localisées dans le « DatabaseHelper », qui contiennent des requêtes SQL préparées et renvoyant des curseurs selon le type d'opération. On cite comme exemple l'enregistrement d'une nouvelle ligne :

```
DBProfileImport DB = new DBProfileImport(ctx, getFilesDir().getAbsolutePath());
DB.insertData(DB, sname, smed, smed1, smed2, smed3, smed4);
Toast.makeText(getBaseContext(),"Add succesful", Toast.LENGTH_SHORT).show();
finish();
```

Il faut noter qu'en cas d'utilisation du curseur, il faut fermer la base de données avec la commande « *closeDatabase() ;* » puisqu'elle n'est pas faite à l'intérieur de la méthode.

III. Développement du GUI et explication par différents scenarios.

Le GUI (Graphic User Interface) est l'interface graphique par laquelle l'utilisateur va interagir.

Plusieurs normes et règles sont nécessaires à l'ergonomie et la simplicité d'utilisation c'est pour lequel on a essayé de faire une interface aussi intuitive que possible tout en gardant un côté esthétique agréable et non encombré.

La conception du GUI peut être très complexe si on souhaite faire une interface attirante ce qui la rend une spécialité en soi vu qu'on utilise des programmes externes.

Dans notre application, le développement du GUI s'est fait en deux parties :

1- La conception des boutons, images et fonds par Adobe Illustrator, puis dépôt des images dans le dossier « Drawable »
2- La création et le paramétrage des layouts XML.

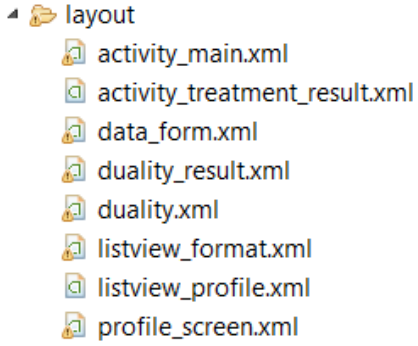


Fig IV.8 Liste des Layouts

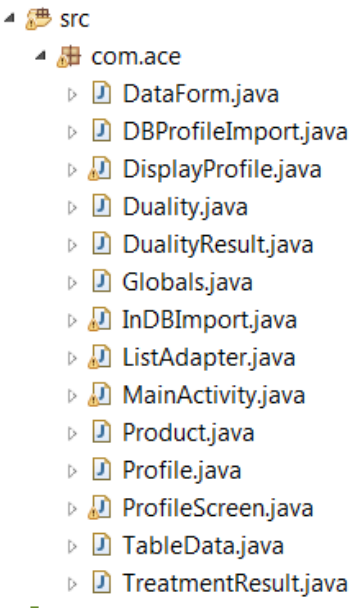


Fig IV.9 Liste des Classes Java

On notera aussi l'utilisation d'une police de texte personnalisée qu'on a importé depuis le dossier « Assets » puis l'appliqué pour chaque élément par la commande suivante :

```
Typeface font = Typeface.createFromAsset(getAssets(), "Chantelli_Antiqua.ttf");
txt.setTypeface(font);
```

On présente ci-dessous le scénario d'utilisation de l'application :

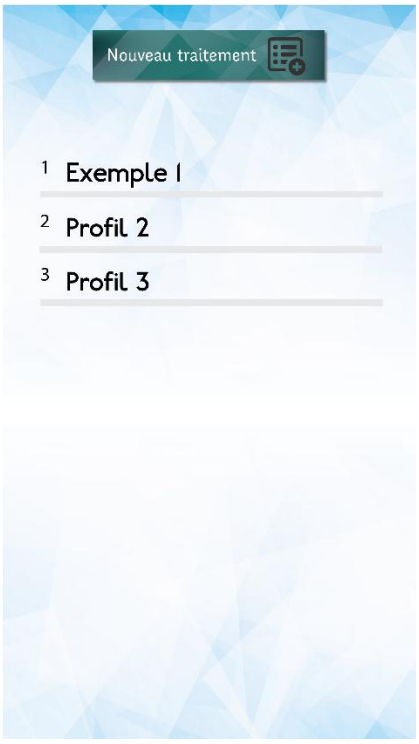
Le lancement de l'application affichera l'interface principale qui correspond à « MainActivity » :

Chaque bouton ouvrira l'interface appropriée, dont on a vu la partie du « Scanner » dans la première partie de ce chapitre.

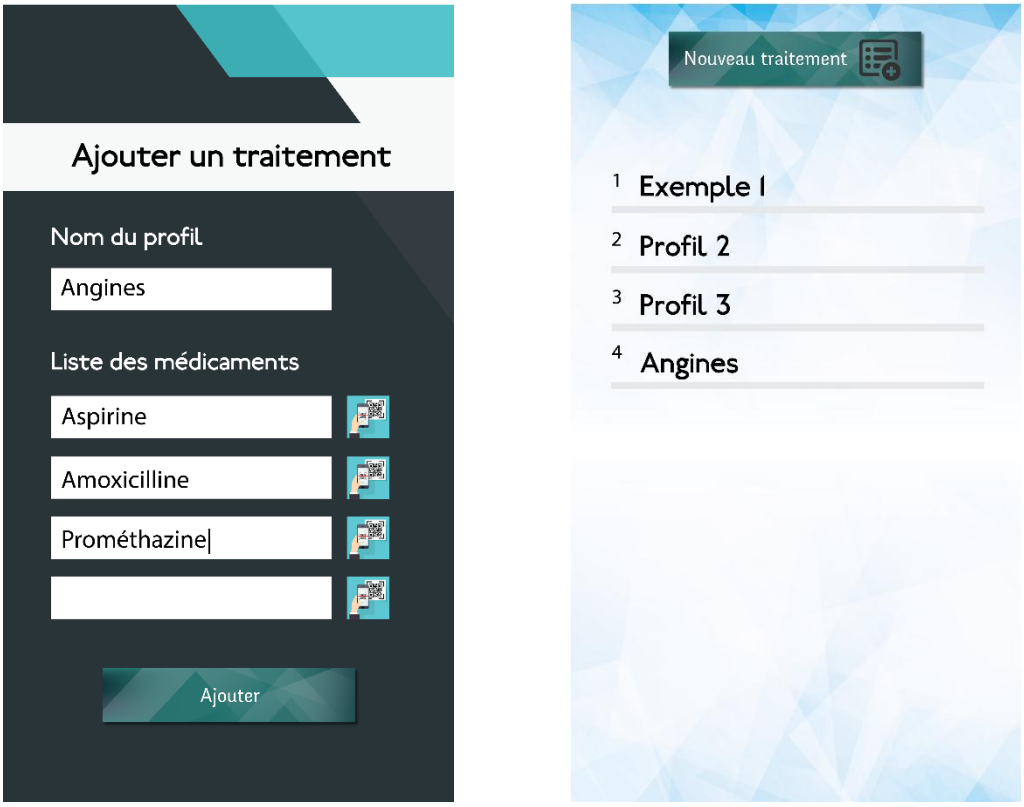
Dans un premier temps, on présentera un exemple de traitement qu'on va tester avec un médicament :

Le traitement contre des angines et une fièvre : Aspirine (Aspegic), antibiotiques (Clamoxyl), Rhinathiol sirop (Prométhazine). Suite à une hypertension, on prescrit au même patient un Antihypertenseur (Atabek).

En appuyant sur la touche profil de traitement on obtient la liste des profils enregistré avec un bouton pour ajouter d'autres qui va nous guider vers une interface à remplir et enregistré un nouveau traitement repérable par son nom de profil :

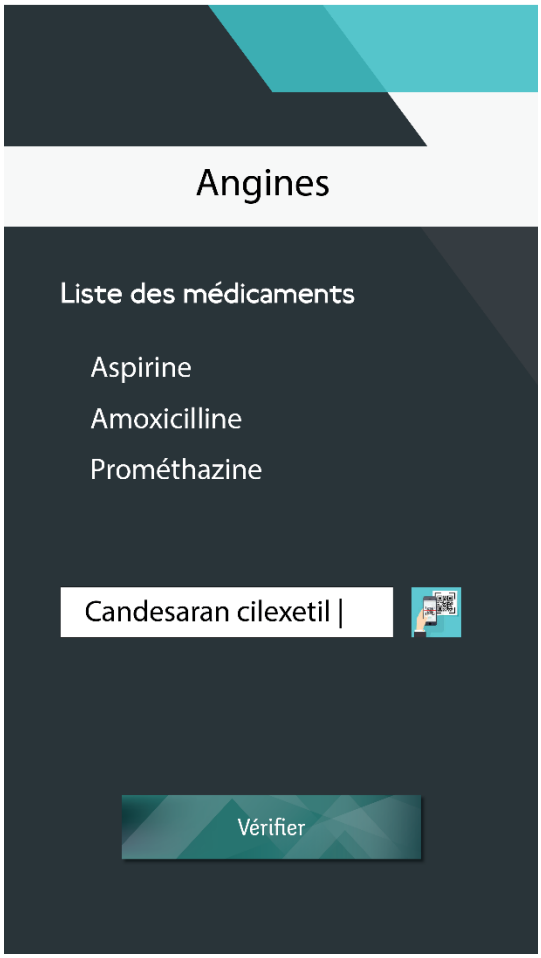


Les noms de profils sont extraits depuis la base de données « PDB » et affichés dans une « ListView » qu'on peut glisser et cliquer sur le contenu. Cette opération est assuré par plusieurs classes et layouts supplémentaires qu'on nomme des « Adapter » avec le seule rôle d'adapter le contenu vers l'affichage. Elle est assez délicate et qui cause la fermeture de l'application si elle est mal implémentée. Par la suite, après qu'un enregistrement soit effectué, son nom de profil sera ajouté à la liste :



Les boutons à coté les champs à saisir servent à entrer automatiquement le CDI du médicament en ouvrant le « Scanner » mais vu la non normalisation du format du code QR notamment en ce pays, il est très difficile en ce moment d'extraire le CDI par un simple scan.

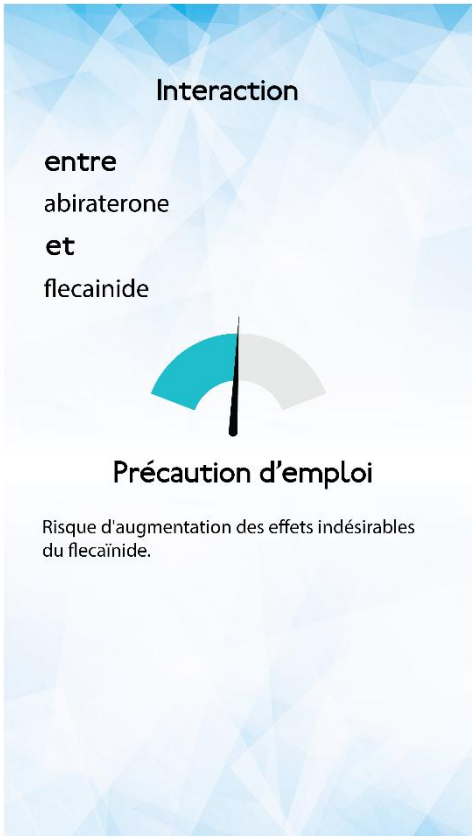
En cliquant sur un nom de profil dans la *listview* l'interface associée au profil sera affichée, avec le champ du médicament à vérifier dans lequel le CDI d'Atabek est entré (son CDI est obtenu depuis l'exemple du scan effectué à la première partie du chapitre) en cliquant le bouton vérifier on obtient le résultat :



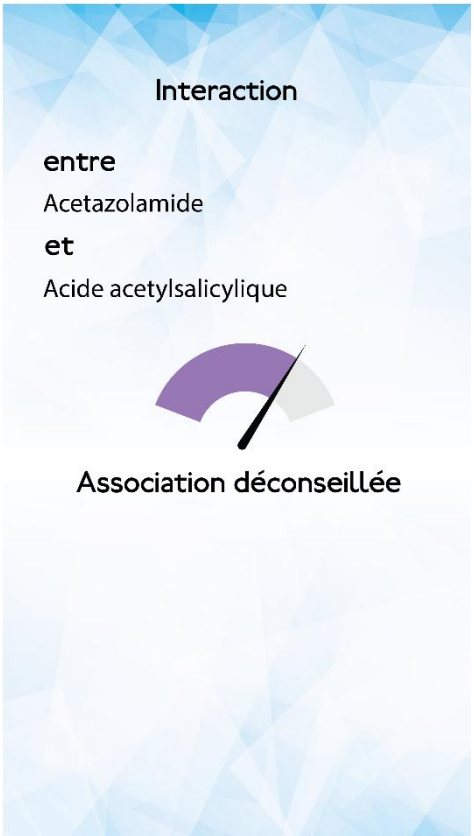
Les informations du résultat sont obtenu après traitement de la base de données « InterDB ». Les détails de l'interaction ne sont pas disponible pour toutes, mais en premier plan, on affiche sa sévirité avec les CDI interagissants.

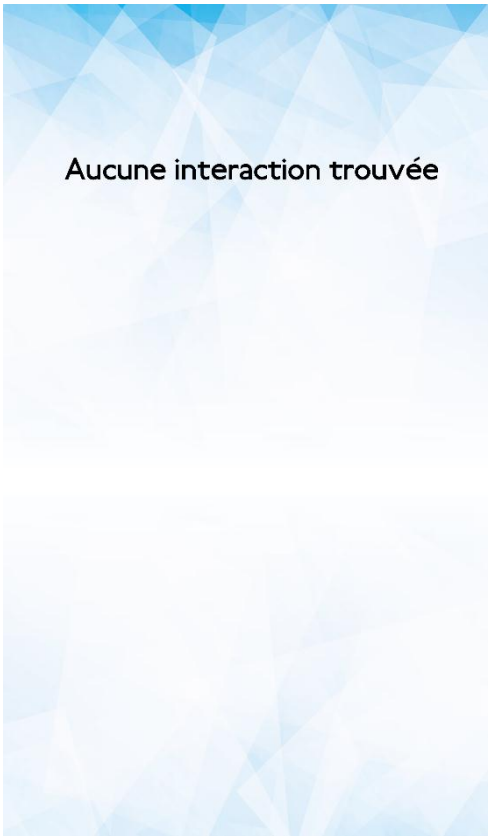
Dans un autre scénario où on veut vérifier deux médicaments, l'opération est assez linéaire et ne nécessite pas l'implication de la « PDB » .

En cliquant sur le deuxième bouton dans la fenêtre principale, puis en entrant les médicaments concernés et validation on obtient le résultat correspondant :



La page du résultat est similaire à celui du traitement. On cite des exemples dans le cas où les détails d'interactions ne sont pas présents ou encore pas d'interaction.





Conclusion :

On a presenté brièvement dans ce chapitre le processus du développement de notre application en présentant le principe d'intégration du lecteur de code-barres QR ainsi que l'implementation des différentes bases de données necessaires aux différentes fonctionnalités de l'application, puis, on a expliqué un peu le développement des interfaces et la conception du GUI. Cependant, on n'a pas pu terminer toutes les fonctionnalités (Principalement l'insertion automatique des médicaments par un simple scan) de notre application dans si court lapse de temps car l'aboutissement d'un tel project nécessite généralement une équipe divisée par chaque tâche en prenant en considération la profondeur et la compléxité de chacune d'elle.

Conclusion

Notre travail s'est basé sur le développement d'un programme sur les appareils mobiles (Smartphone, tablette, ...), ceci nous a amené à mieux connaitre la plateforme de développement et à enrichir notre savoir et notre expérience.

L'application que nous avons réalisée permet à une personne suivant déjà un traitement médical spécifique à une altération de la santé, d'évaluer le niveau de l'interaction médicamenteuse si ce malade venait à ajouter un autre médicament à son traitement. Alors cette application pourra gérer les différentes thérapies suivies pas le malade en s'appuyant sur les médicaments qu'il prend.

Les objectifs fixés de notre projet de fin d'études ont été atteint à un degré très avancé. D'un côté nous avons présenté une application bénéfique aussi pour les malades que pour le corps médical (médecin, pharmacien). D'un autre côté, ce projet nous a permis de se perfectionner avec Android et d'approfondir nos connaissances dans le domaine des applications mobiles.

Dû à l'étroite période allouée pour développer notre projet, il nous était impossible d'achever tous les fonctionnalités et la finaliser c'est pour cela que d'éventuelles perspectives d'imposent :

- Enrichir la base de données avec plus d'interactions.
- Ajouter les interactions avec les substances.
- Finaliser le saisi automatique des médicaments par scan de code QR.
- Ajouter un rappel des temps de prises des médicaments enregistrés dans un traitement.

Liste des figures

Liste des tableaux

Références bibliographiques

[1] Scheen A. Luyckx A. Médicaments et maladies iatrogènes. Revue Medicale Liège,2005. 210,page 61-69.

[2] Stockley. Drug interactions. The pharmaceutical Press, London, 2002,1200, page 1080 .

[3] Sandson N. Drug-drug interactions: the silent epidemic. Psychiatric Services, 2005,360, page 22-24.

[4] Diquet B, Laine-Cessac P. The search for risk factors and the therapeutic consequences. Drug interactions. Revue Prat, 2005, page 541-545.

[5] Afssaps. Commission nationale de pharmacovigilance du 25 mars 2008.Etude EMIR (effets indésirables des médicaments : incidence et risque) sur les hospitalisations liées à un effet indésirable médicamenteux. 20 mai 2008. page 21.

[6] Arques-Armoiry E, Cabelguenne D, Stamm C. Problémes médicamenteux les plus fréquemment détectés par l'analyse pharmaco thérapeutique des prescriptions dans un CHU. Revue Med Interne 2010, page11.

[7] Thèse : Nathalie MATHIEU. Interactions médicamenteuses : De la théorie à la réalité 2010. UNIVERSITE HENRI POINCARE - NANCY 1 .Faculté de pharmacie. 147. 14-19 pages.

[8] Siobhan Dumbreck , Angela Flynn, Moray Nairn, The British Medical Journal , Publication Mars 2015

[9] Association des pharmaciens du Canada. Compendium des produits et spécialités pharmaceutiques. Ottawa : Association des pharmaciens du Canada : 2005

[10] Huang Xuguang, An Introduction to Android , Database Laboratory. Novembre 2011.

[11] Paweł Bedynski, Andood – an Android application, Master's thesis in COMPUTER SCIENCE, University of Warsaw - Faculty of Mathematics, Informatics and Mechanics. Juin 2011.

[12] NHS Innovations South East, App Development: An NHS Guide for Developing Mobile Healthcare Applications . Mai 2014.

[13] Benny Skogberg : Android Application Development – A Guide for the Intermediate Developer , Malmö University , Suède. September 2010.

[14] Ji Qianyu, Exploring The Concept Of Qr Code And The Benefits Of Using Qr Code For Companies, School of Business and Culture Degree Programme in Business Information Technology. 2014.

[15] Libby Kirkland, Area & Perimeter, Instructional Technology Coach. 2012.

[16] Mike Wolfson, Donn Felker : Android Developer Tools Essentials: Android Studio to Zipalign . "O'Reilly Media, Inc.", 14 août 2013.

[17] Sébastien Pittion, Bastien Siebman : Applications mobiles avec Cordova et PhoneGap. Edition EYEROLLES. Paris Cedex 2014.

[18] Barry Burd : Android Application Development All-in-one for Dummies. Jhon Wiley & Sons Inc. 2012.

[19] Damien Guignard, Julien Chable, Emmanuel Robles : Programmation Android, De la conception au déploiement avec le SDK Google Android. Edition EYEROLLES. Paris Cedex 2015.

[20] Bruno Mermet , Construction et gestion de développement avec Maven 3.0, Novembre 2010.

[21] Dominique A. Heger : Mobile Devices - An Introduction to the Android Operating Environment , Design, Architecture, and Performance Implications. 2011.

[22] SCHMIDT, Aubrey-Derrick ; BYE, Rainer ; SCHMIDT, Hans-Gunther ; CLAUSEN, Jan H. ; KIRAZ, Osman ; Y¨UKSEL, Kamer A. ; C¸ AMTEPE, Seyit A. ; ALBAYRAK, Sahin: Static Analysis of Executables for Collaborative Malware Detection on Android. In: ICC, 2009, S. 1–5.

[23] SHABTAI, A. ; FLEDEL, Y. ; KANONOV, U. ; ELOVICI, Y. ; DOLEV, S.: Google Android: A State-of-the-Art Review of Security Mechanisms. In: Arxiv preprint arXiv:0912.5101 (2009).

[24] Mathias Séguy : Android, A Complete Course - From Basics to Enterprise Edition. Android2ee , 2011.

Lexique

ADT	Android Development Tool, Plugin d'outils de développement d'Android.
Action thérapeutique	Manière de traiter plusieurs maladies.
Antagonistes	Substance qui empêche une autre d'agir correctement chez un être vivant.
API	Interface Applicative de Programmation pour établir des connexions entre plusieurs logiciels.
Classe Java	Déclare des propriétés communes à un ensemble d'objets.
Conséquences cliniques	Signe détecté au simple examen, par la vue et le toucher.
Effets asymptomatiques	Maladie sans signes apparents.
GUI	Graphical User Interface , interface graphique d'utilisateur.
Hypotension artérielle orthostatique	Une diminution de la pression du sang lors du passage de la position couchée ou assise, à la position debout
Le principe actif	Molécule entrant dans la composition d'un médicament et lui conférant ses propriétés thérapeutiques. Un médicament contient un ou plusieurs principes actifs incorporés dans un excipient
ListView	Mode d'affichage en liste avec éléments interactifs.
Layout	La mise en page de l'interface d'utilisateur.
Métabolisme	Transformation du principe actif du médicament dans un corps humain.
Neuromédiateur	Substance chimique servant de relais à la transmission de l'influx nerveux.
OS	Operating System : Système d'exploitation.
Paracétamol	Principe actif d'usage courant, utilisé comme analgésique (contre la douleur) et comme antipyrétique (contre la fièvre).
Plugin	Petit logiciel qui se greffe à un programme principal pour lui conférer de nouvelles fonctionnalités
SDK	Software Development Kit, ensemble d'outils d'aide à la programmation proposé aux développeurs.
Système physiologique	Le mode de fonctionnement d'un corps d'être vivant.