الجمهورية الجزائرية الديمقراطية الشعبية
République Algérienne Démocratique et Populaire

Ministère de l'Enseignement Supérieur
et de la Recherche Scientifique
Université Akli Mohand Oulhadj - Bouira -
Tasdawit Akli Muḥend Ulḥağ - Tubirett -

وزارة التعليم العالي والبحث العلمي
جامعة أكلي محند أولحاج
-البويرة-

جامعة البويرة

Faculté des Sciences et des Sciences Appliquées

كلية العلوم والعلوم التطبيقية

Référence : ...../MM/2024

المرجع:...../م م/ 2024

Mémoire de Master

Présenté au

Département : Génie Électrique
Domaine : Sciences et Technologies
Filière : Electronique
Spécialité : Electronique des systèmes embarqués

Réalisé par :

FOUDI Yacine

Et

BOUKHELF Moussa

Thème

Amélioration de l'application pour le diagnostic et le Suivi de la Faculté des Sciences et des Sciences Appliquées

Soutenu le: **10/07/2024**

Devant le Jury composé de :

Mr :	TOUAFEK MED YAAKOUB	M.A.A	Univ. Bouira	Président
	NOURINE MOURAD	M.C.A	Univ. Bouira	Rapporteur
	CHEBI HOCINE	M.C.B	Univ. Bouira	Examinateur

Dédicaces :

Grace a l'aide de Dieux

Je dédie ce modeste travail en signe de respect

Je le dédie :

A mes parents qui m'ont apporté leur soutien et leur aide financière . sans eux je n'aurais jamais pu atteindre la fin de mes études .

A ceux qui m'ont soutenu durant les moments difficiles de mon parcour

A mes deux frères ,j'espère qu'ils réaliseront tout ce qu'ils souhaitent

A tout ma famille , sans exception

A tous mes amis

A ma binôme « Moussa Boukhelf » et à toute sa famille.

Toute la promo de Electronique des Systèmes Embarquées 2023 /2024

Yacine

Dédicaces :

Je dédie ce modeste travail en signe de respect et d'amour : Je le dédié : A mes parents qui m'ont soutenu dans la vie et qui m'ont accompagnée Dans mon parcours.

A ceux qui m'ont soutenu dans les étapes les plus importantes.

A toute ma famille, sans exception.

A ma binôme « Yacine Foudi» et à toute sa famille.

À tous mes amis.

A tous ceux dont j'ai oublié de mentionner les noms.

Toute la promo de Electronique des Systèmes Embarquées 2023/2024.

Moussa

Remerciements

Ce travail a été effectué au sein du Département des Sciences et sciences appliquées de l'Université de Bouira.

Nous tenons à remercier Allah le tout puissant qui nous a accordé santé et courage pour mener ce travail jusqu'à son terme.

Nous tenons à remercier également notre promoteur ***Mr. NOURINE Mourad*** *qui a accepté de nous encadrer et qui nous a toujours guidés dans la réalisation de ce mémoire.*

Sincères remerciement aux membres de jury.

Nous remercient tous notre enseignants, nous tiens à leur exprimer notre reconnaissance pour avoir accompagné tout au long de notre formation.

Enfin, j'associe à ces remerciements tous ceux qui ont contribué à réaliser ce travail.

Résumé

L'objectif de ce PFE est la mise à disposition du service maintenance de la FSSA de l'université de Bouira d'une application informatique , pour le soutenu de ce service à la gestion, le suivi quotidien et le diagnostic des contrainte rencontrées sur le terrain de la FSSA : fuite d'eau , fenêtre cassées , bureaux endommagé

Cette application est embarquée sur , des téléphones smartphones dont le système d'exploitation android ,Elle peut aussi être installée sur des pc (portable ou bureau) .

L'exploitation de cette application va permettre à terme gestion efficace , un suivis permanent des problèmes signalés eu temps réel , dont l'impact sera visible au niveau de la prise en charge des problèmes soulevés , ce moyen informatique moderne va permettre également une amélioration dans la gestion d'une partie des problèmes signalés de la faculté , et d'aider donc à une meilleure qualité du services rendus

Sommaire

Liste des figures :

Chapitre I : Généralités sur les applications et les systèmes d'exploitation

Chapitre II : Les outils et logiciels utilisée

Chapitre III : Développement de l'application et résultats obtenus

Liste des tableaux :

Liste des abréviations :

HTML Hyper Text Markup Language .

OS Operating System

iOS Inter Network Operating system

JDE Environnement de Development Java

SDK Software development kit

XML Extensible markup language

IDE Integrated development environment

Iso International Organization for Standardization

St Science Technic

3D three-dimensional

FSSA Faculté des sciences et des sciences appliquées

Introduction Générale

La FFSA est l'une des Facultés de l'Université de Bouira, crée depuis maintenant quelques années seulement, elle est donc récente en terme de construction architecturale. Cependant, de nombreux problèmes sont quotidiennement enregistrés par des agents habilités au service de la maintenance de la Faculté : fuites d'eau, vitres cassées, lampes grillées, prises d'électricités arrachées, portes de bureaux défoncées, problèmes d'étanchéités de quelques salles et amphis, robinets d'eau, …. Etc. La liste de ces petits problèmes est ici loin d'être exhaustive, mais elle est clairement indicative.

À titre d'exemple, lors d'un signalement d'une fuite d'eau, dont les endommagements n'est absolument pas à sous-estimer, l'endroit de la fuite signalée doit être bien localiser sur le vaste site de la Faculté : salle de cours, amphi, laboratoire de recherche, couloir de l'étage, sanitaire, bureau de …., etc. Ce signalement s'effectue actuellement par les agents de maintenance d'une manière classique, c'est-à-dire : un rapport version papier, et parfois même un rapport verbal. Le suivi de ces rapports et la réactivité de ce service aux multiples problèmes signalés (la gestion) devient difficile, à cause principalement du caractère urgent (ou non) à y remédier, pour préserver et sauvegarder l'image du fonctionnement de la Faculté.

L'objectif principal de ce PFE est de proposer une solution électronique et informatique, pour la gestion et le suivi de l'ensemble des dysfonctionnements cités précédemment. Elle consiste à établir des liens modernes de communications, via des Smartphones et des PC, entre les agents de terrain et le service de maintenance de la Facultés. La traçabilité électronique des problèmes signalés va permettre certainement une gestion efficace de ceux-ci, et de diagnostiquer rapidement les récurrents problèmes rencontrés, pour une meilleure prise en charge. L'usage des moyens technologiques est une valeur ajoutée sûr pour améliorer le cadre de vie au sein de notre Faculté.

Ce manuscrit est organisé en trois (3) chapitres :

- Le premier chapitre est dédié a les généralités sur les applications mobiles et les systèmes d'exploitation
- Le deuxième chapitre est consacré sur les outils et logiciels utilisé dans notre PFE
- Au dernier chapitre sont exposés et présentés les résultats obtenus et le développement de notre application

En fin, nous terminons ce modeste travail par une conclusion générale.

Chapitre I :

Généralités sur les applications mobiles et les systèmes d'exploitation

Chapitre I : Généralités sur les applications et les systèmes d'exploitation

I.1 Introduction :

Au cours des dernières années, le marché des téléphones mobiles a connu une croissance rapide, et presque tout le monde possède aujourd'hui un téléphone ou une tablette , Avec les avancées technologiques, les téléphones portables ne se limitent plus aux appels et aux jeux. Par exemple, , grâce aux applications mobiles les smartphones nous permettent de planifier notre journée, de consulter nos e-mails, de participer à des visioconférences, de nous connecter aux réseaux sociaux

Dans ce chapitre, nous allons examiner la notion d'application mobile, ses diverses catégories et caractéristiques. Nous couvrirons également les systèmes d'exploitation, les outils de développement mobile, en particulier Android Studio, ainsi que les langages de programmation utilisés, comme 'Dart'.

I.2 Les applications mobiles :

Le terme « application mobile » désigne un logiciel. Il s'agit plus précisément d'un programme, qui contient un fichier, pouvant être téléchargé depuis un téléphone mobile ou encore, depuis une tablette. Une fois installé, ce programme est exécuté par le système d'exploitation du smartphone ou de la tablette sur lequel il se trouve. **[1]**.

I.2.2 Les types d'application mobiles :

Selon les fonctionnalités et les usages qu'on leur confère, il existe différents types d'applications mobiles **[2].**

a. **Les applications Natives :** la plupart des applications que nous téléchargeons sur nos mobiles aujourd'hui sont dites natives. L'application mobile native est développée pour un système d'exploitation en particulier (IOS ou Android). Elles utilisent généralement les fonctionnalités existantes sur nos mobiles pour proposer une expérience utilisateur unique.

b. **Les applications web : :** Enfin, la troisième typologie d'application mobile concerne les Web Apps ou autrement dit, les applications web. Ce sont des duplicatas de sites web optimisés pour être utilisés sur mobile. Pour naviguer sur une Web App, on utilise les moteurs de recherche de son mobile. Le point faible de ces applications mobiles, c'est qu'elles ne peuvent pas être utilisées hors connexion internet **[2].**

c. **Les applications hybrides :** Dans l'environnement des applications mobiles, on retrouve également les applications hybrides. On utilise le terme « hybride » parce que ce type d'application mobile combine les éléments des Natives Apps et des Web Apps. Puisqu'elles sont codées à partir de langages web tels que HTML 5 et JavaScript, elles peuvent être téléchargées depuis toutes les plateformes mobiles et c'est ce qui fait leur grande popularité**[2].**

I.3 Les systèmes d'exploitation mobile OS :

Le système d'exploitation est le logiciel qui lui permet de fonctionner. Il contrôle et gère les fonctions essentielles de l'appareil. Aussi, il sert de lien entre l'utilisateur et son smartphone et lui permet donc de profiter de ses fonctionnalités**[3].**

I.3.2 Les principaux systèmes d'exploitation mobiles :

a. **Android (Google) :** Est le système d'exploitation le plus répandu. Développé par Google, il n'est pas rattaché à une marque de smartphones (contrairement à iOS

Les fabricants intègrent Android à leurs smartphones et y apportent quelques modifications. Certaines applications sont cependant intégrées automatiquement : Google, Gmail, Google Maps, YouTube.

Figure I. 1 : Logo Android **[4].**

b. **IOS (IPhone) :** Après Android, iOS est le deuxième système d'exploitation le plus répandu. Il équipe uniquement les « iPhone », c'est-à-dire les smartphones de la marque Apple.Le premier iPhone fut commercialisé en 2007. Depuis, plusieurs modèles sont sortis, généralement accompagnés d'une nouvelle version du système d'exploitation. Par exemple, fin 2014, la sortie de l'iPhone 6 a coïncidé avec l'arrivée de l'iOS 8**[4].**

Figure I. 2 : Logo IOS **[4].**

c. **Windows Mobile :** Windows 10 Mobile (anciennement Windows Phone) est un système d'exploitation développé par Microsoft que l'on retrouve sur plusieurs marques comme Nokia, HTC et Acer ,ce système d'exploitation est plus récent que ses concurrents (Android et iOS), ceci explique par exemple que son store contient moins d'applications à l'heure actuelle.

La version 10 est assez proche du système d'exploitation disponible sur ordinateur (Windows 10), **[4].**

Figure I. 3 : Logo Windows **[4].**

d. **BlackBerry OS :** BlackBerry OS est probablement le moins connu des systèmes d'exploitation présentés ici. On ne le trouve que sur les smartphones de la marque BlackBerry. Chaque version de BlackBerry s'appelle « OS » + le numéro de la version. En 2017, c'est la version OS 10.3.2 qui est généralement installée sur les nouveaux Appareils, Le store de BlackBerry (BlackBerry App World) compte beaucoup moins d'applications que les leaders du marché (App Store et Google Play Store)**[4].**

Figure I. 4 : Logo BlackBerry **[4].**

I.3.3 Comparaison entre les Systèmes d'exploitation :

	IOS	*ANDROID*	*WINDOWS*	*BLACKBERRY*
Langage de programmation	Ojective-c	java	C ; C++	java
Disponibilité de l'environnement de développement	X code	Android studio , Eclipse	Visuel studio	JDE
Platform	Iphone, IPad	Android (samsung , google)	Windows mobile	Blackberry
Magasin en ligne	App Store	Google play store	Windows play Store	App world
Open source	NON	OUI	NON	OUI
Créateur	Apple	Google	Microsoft	RIM

Tableau I. 1 : Comparaison entre les Systèmes d'exploitation **[4].**

I.4 Le système d'exploitation Android :

Un système d'exploitation est le logiciel central qui permet à un appareil de fonctionner en contrôlant et en gérant ses principales fonctions.

I.4.1 Le choix de la plateforme Android :

Le système d'exploitation Android est totalement Open Source. Cela veut dire que quiconque, même les concurrents d'Android, peut choisir de télécharger, d'installer, de modifier et de distribuer son code source sans frais. Grâce à Android, jamais autant de personnes n'ont eu accès à la puissance de la technologie mobile **[5].**

I.4.2 Naissance d'Android :

La création d'Android remonte à 2003 avec une société américaine dénommée ANDROID. Deux ans plus tard, donc en 2005, Google rachète Android tout en gardant sa renommée. A cette époque son objectif était de développer un système d'exploitation facile et compréhensif pour tous. Avec BugDroid, Android a conquis presque le monde entier **[6].**

I.4.3 Les Version De La plateforme Android :

Version Android	Nom	Date de Lancement	Logo
1.0	Apple Pie	23 septembre 2008	 Android apple pie
1.1	Banana Bread	9 février 2009	 Android Banana Bread
1.5	Cupcake	30 avril 2009	 Android Cupcake
1.6	Donut	15 septembre 2009	 Android Donut
2.0	Éclair	26 octobre 2009	 Android Éclair
2.2	Froyo	20 mai 2010	 Android Froyo
2.3	Gingerbread	6 décembre 2010	 Android Gingerbread
3.0	Honeycomb	22 février 2011	 Android Honeycomb
4.0	ICE Cream Sandwich	19 octobre 2011	 Android ICE Cream Sandwich
4.1/4.2/4.3	Jelly Bean	9 juillet 2012	 Android Jelly Bean
4.4	KitKat	3 septembre 2013	 Android KitKat
5.0	Lollipop	15 octobre 2014	 Android Lollipop
6.0	Marshmallow	28 mai 2015	 Android Marshmallow
7.0	Nougat	10 mars 2016	 Android Nougat

Tableau I. 2 : Historique des versions Android **(2008 -2016) [6].**

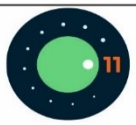
Version Android	Nom	Date de Lancement	Logo
8.0	Oreo	21 août 2017	
9.0	Pie	7 Mars 2018	
10	Q	3 septembre 2019	
11	R	19 février 2020	
12	S	18 février 2021	
13	T	10 février 2022	

Tableau I. 3 : Historique des version Android **(2016 - 2022) [6].**

I.4.4 Le Kit de développement (SDK) :

Un kit de développement logiciel (SDK) est un ensemble d'outils de création spécifiques à une plateforme destinés aux développeurs. Le besoin de composants tels que des débogueurs, des compilateurs et des bibliothèques pour créer du code qui s'exécute sur une plateforme, un système d'exploitation ou un langage de programmation spécifique. Les kits SDK regroupent tout les besoins pour développer et exécuter des logiciels en un seul endroit. En outre, ils contiennent des ressources telles que de la documentation, des didacticiels et des guides, ainsi que des API et des cadres pour accélérer le développement d'applications **[7].**

I.5 Conclusion :

Ce chapitre explore les différentes catégories et caractéristiques des applications mobiles, les systèmes d'exploitation qui les supportent, et les outils de développement mobile, notamment Android Studio et le langage de programmation Dart. L'ensemble des outils introduits dans ce chapitre sera utilisé lors de la création de notre application .

Chapitre II :

Les outils et logiciels utilisées

Chapitre II : Les outils et logiciels utilisées

II.1 Introduction :

Dans ce chapitre, nous allons explorer les différents outils et logiciels nécessaires à la conception et au développement de notre application. Les outils utilisés incluent un PC, un téléphone portable et un modem. Nous utiliserons également plusieurs logiciels clés, notamment Android Studio, Visual Studio, ainsi que les langages de programmation Flutter et Dart. Chaque outil et logiciel sera détaillé pour montrer comment il contribue à la réussite de notre projet.

II.2 Android studio :

C'est un environnement de développement pour développer des applications mobiles Android. Il peut être téléchargé sous les systèmes d'exploitation Windows, macOS, Chrome OS et Linux2 **[8].**

II.2.1 Fonctionnalités :

Android Studio permet principalement d'éditer les fichiers Java/Kotlin (langage de programmation) et les fichiers de configuration XML (Extensible Markup Langage) d'une application Android, il intègre par ailleurs un émulateur permettant de faire tourner un système Android virtuel sur un ordinateur **[8].**

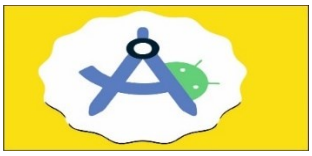


Figure II. 1: Logo Android studio **[8].**

II.3 Langages de développement de notre application :

II.3.1 Dart :

Dart est un langage de programmation optimisé pour les applications sur plusieurs plateformes. Il est développé par Google et est utilisé pour créer des applications mobiles, de bureau, de serveur et web**[11].**

Figure II. 2 : Logo Dart **[11].**

II.3.2 Flutter et dart :

Flutter fonctionne grâce à Dart, un langage conçu pour des applications performantes sur toutes les plateformes. Il permet de déployer une seule base de code sur divers appareils : mobiles, web, ordinateurs de bureau, utilisé par Google et approuvé par de nombreuses marques renommées à l'échelle mondiale, il est soutenu par une vaste communauté de développeurs internationaux**[11].**

Figure II. 3: Logo Dart et Flutter **[11].**

II.4 Création d'application Flutter :

L'ensemble des étapes principales de création d'application Flutter sur Android Studio et résumé dans ce qui suit :

II.4.1 Définition du Flutter :

Flutter est un kit de développement logiciel (SDK) d'interface utilisateur open-source créé par Google. Il est utilisé pour développer des applications sur Android, iOS, Linux, Mac, Windows, Google Fuchsia et le web à partir d'une seule base de code **[9].**

Figure II. 4 : Logo de Flutter **[8].**

Les principales composantes de Flutter sont les suivantes **[9]** :

- Langage de programmation Dart
- Le moteur Flutter
- Les bibliothèques
- Les widgets spécifiques à la conception

II.4.2 Création de l'application Flutter :

Pour créer une application Flutter, les étapes suivantes doivent être suivies dans l'ordre **[10]** :

Etape 1 : Installer Flutter :

Pour commencer à créer une Application Flutter, il faut d'abord installer Flutter **[10]** :

- Ouvrir le lien '' **https://docs.flutter.dev/get-started/install** ''
- Cliquer sur Windows pour choisir le Système d'exploitation approprie (Figure II.5)
- Apres Cliquer sur Desktop pour choisir la machine de travail (Figure II.5)
- En fin cliquer sur télécharger Flutter. (Figure II.5)

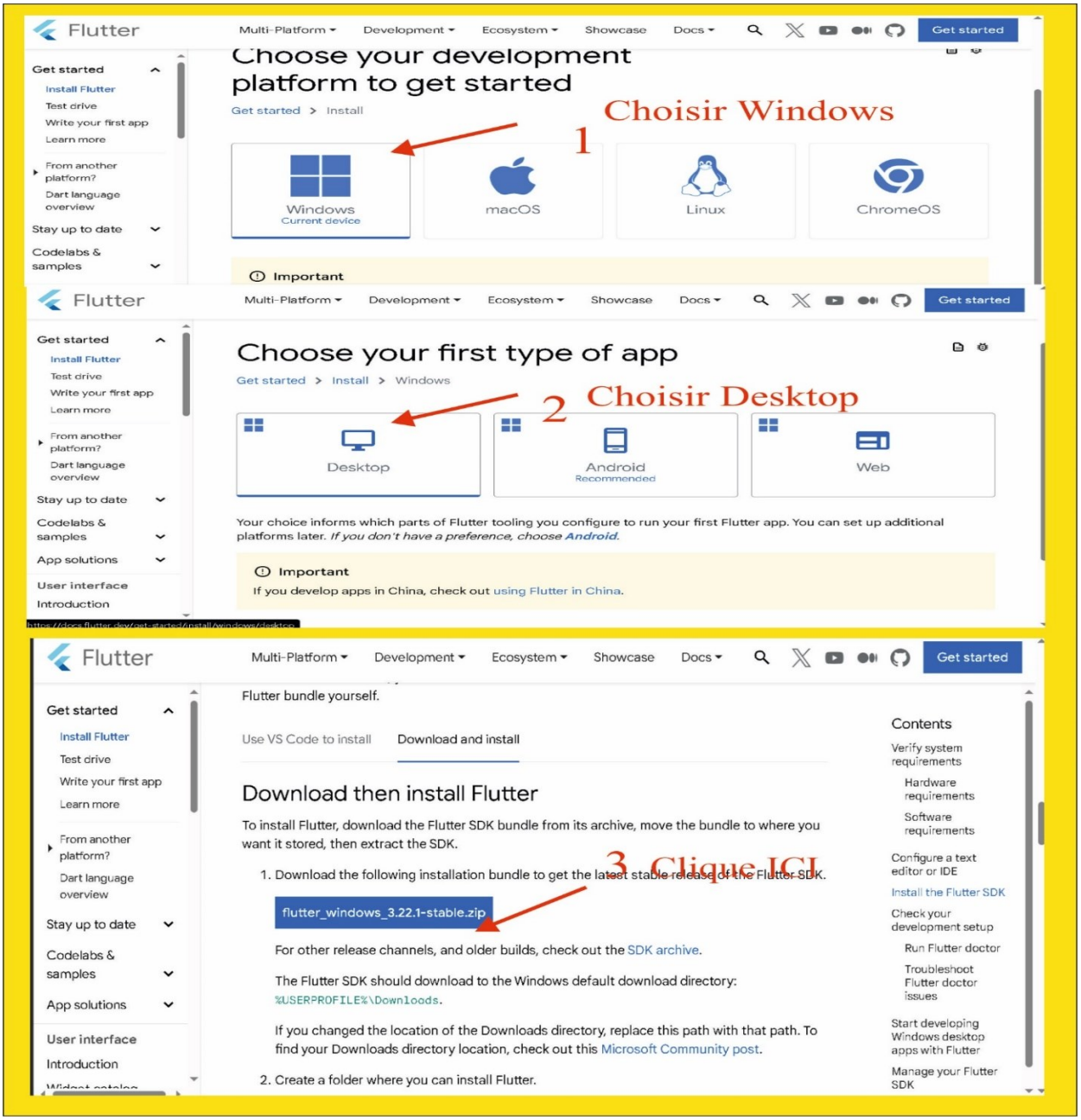


Figure II. 5: Installation de Flutter **[10].**

Etape 2 : Création du projet Flutter :

Cette étape permet de créer un nouveau projet en suivant les étapes indiquées sur la Figure (II.6) :

- Ouverture de l' EDE Android studio' .
- Sélection 'new' , puis 'new Flutter Project'.

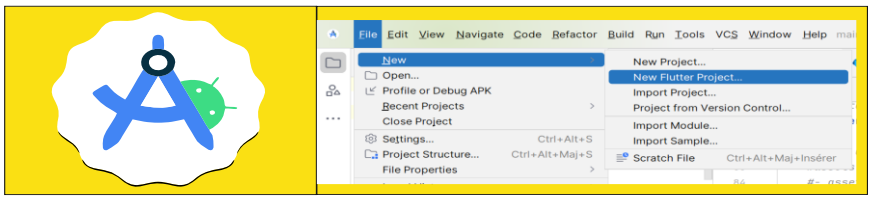


Figure II. 6: Fenêtre de L'IDE Android studio**.**

Etape 3 : Cette étape permet de nommer l'application crée , elle consiste à sélectionner Flutter , puis next , et donner le nom de l'application . Dans notre cas le nom est : 'St-diagnostic' ,ces étapes sont résumées sur (la Figure II.7)

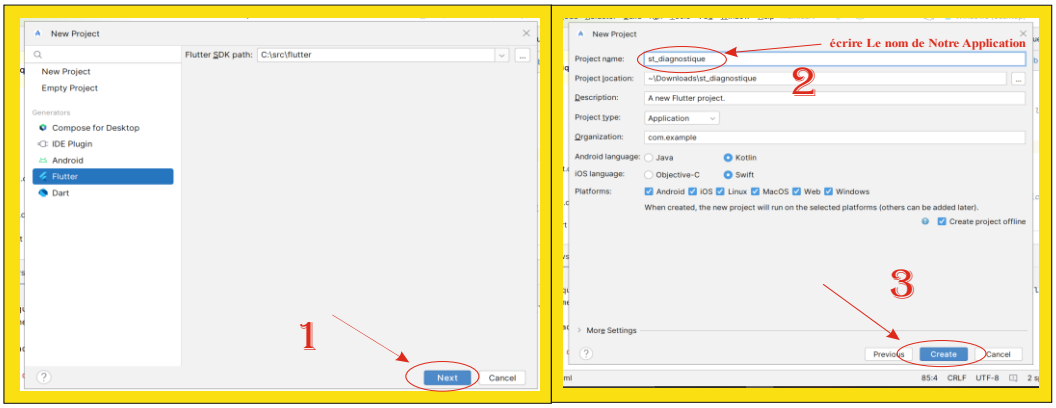


Figure II. 7 : Fenêtre de L'IDE Android studio.

La figure (II.8) illustre la structure du projet ainsi crée :

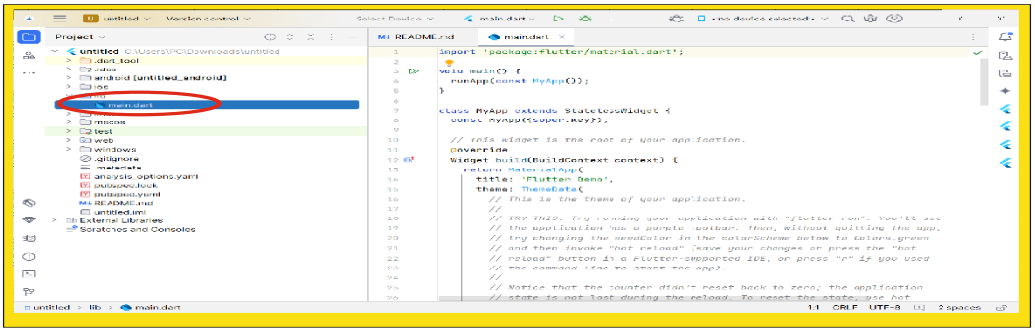


Figure II. 8 : Fenêtre de L'IDE Android studio.

II.4.3 Explication de la structure d'un projet [10] :

Android : le dossier génère automatiquement le code pour l'application d'Android.

iso : le dossier génère automatiquement le code pour l'application d'ios.

lib : le dossier d'accueil contient le code Dart de l'application.

lib/main.dart : le fichier est convoqué pour démarrer (Start) l'application.

test : le dossier contient les codes Dart pour tester l'application.

test/widget_test.dart: simple code.

Pubspec.lock : ce fichier doit être ajouté à GIT Control pour s'assurer que les membres de votre équipe de développement utilisent les mêmes versions de bibliothèque.

README.md : Le fichier décrit le projet, lequel est écrit selon la structure Markdown.

II.4.4 Ecriture du code pour l'application :

Cette étape consiste à supprimé le code initial et le remplacé avec le code de l'application proposée (voir la Figure II.9) .

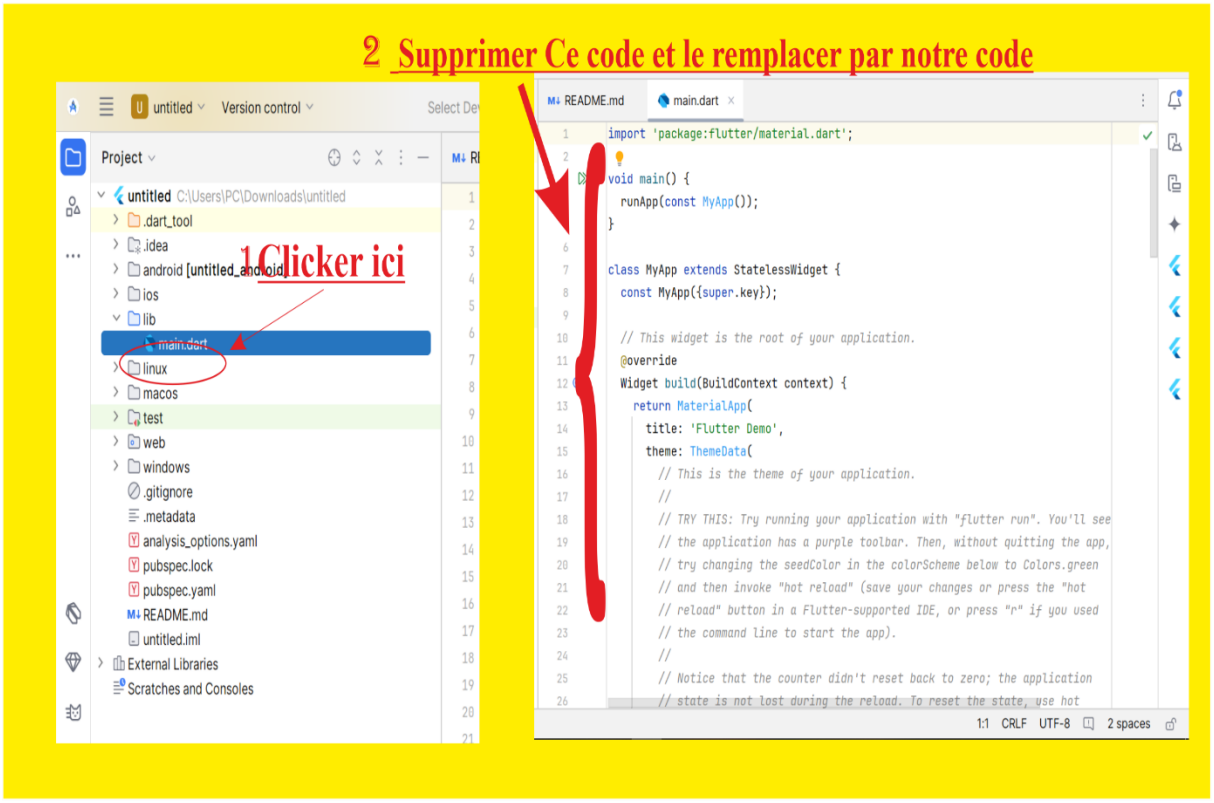


Figure II. 9 : Le code Initial.

Exemple de code d'application utilisée :

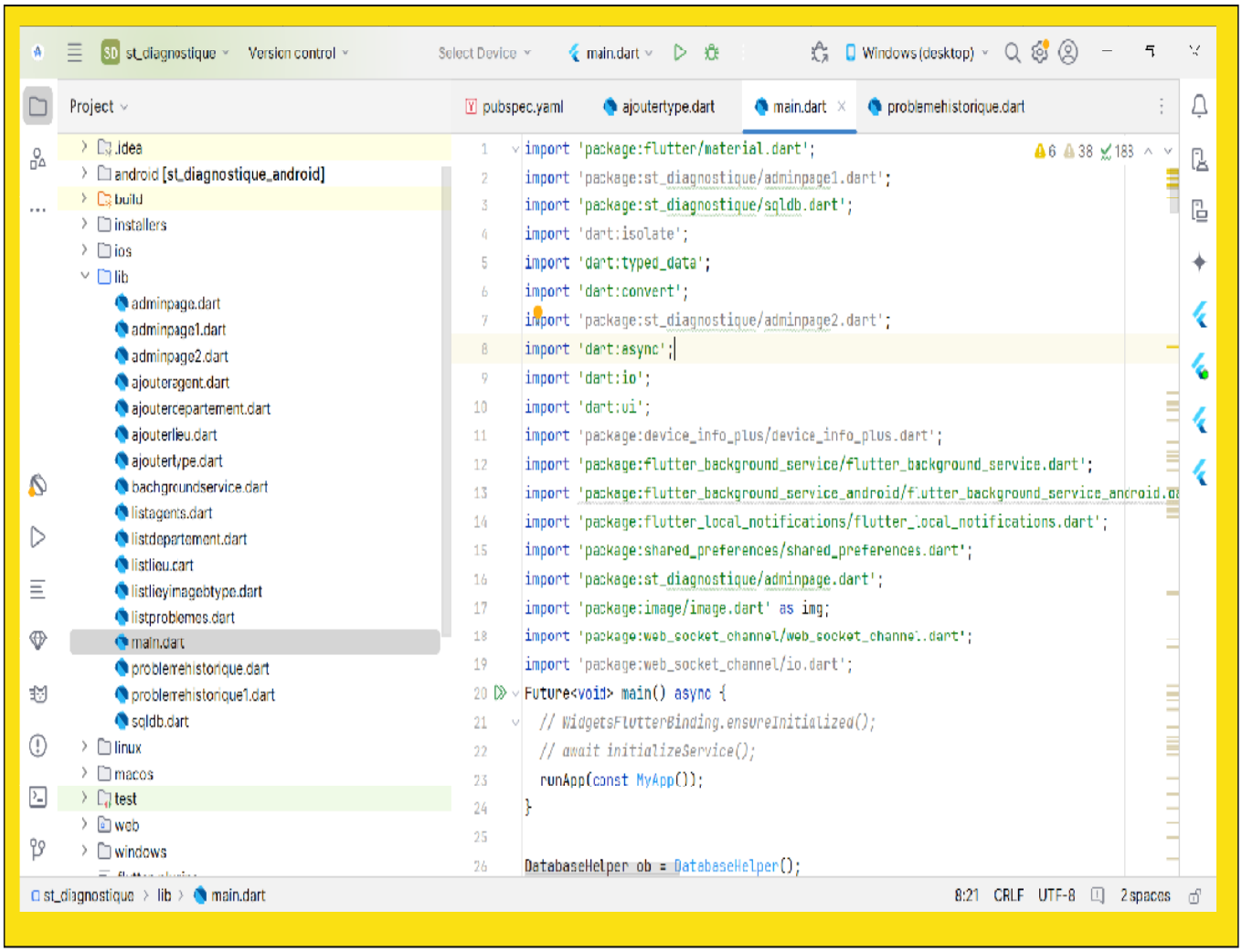


Figure II. 10 : Exemple du code de notre application .

II.4.5 Lancement de l'application :

L'application Flutter doit être mise en place dans un appareil d'Android, par conséquent, au cours du processus de développement, nous avons comme choix l'une des options suivantes :

- Sélectionné nouveau Appareil Android (New device selected) .
- Connecter l'appareil à l'ordinateur , et activer le mode développeur (developper mode).
- Exécuter un émulateur Android (Android Emulator) , comme illustre la figure (II.11)

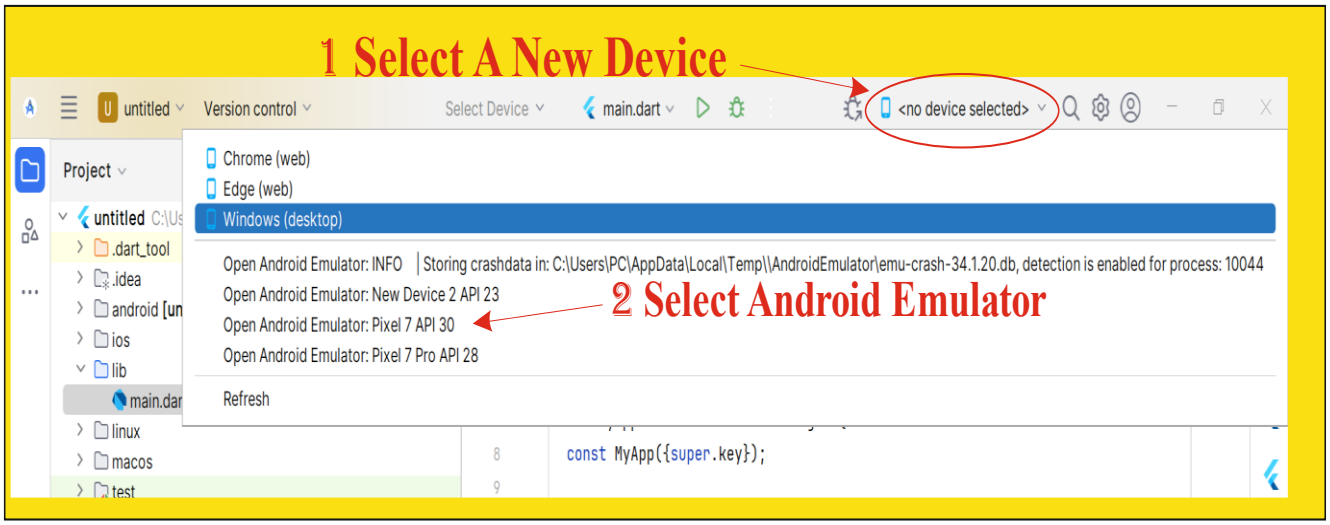


Figure II. 11 : Fenêtre de sectionnement de l'émulateur.

Après avoir sélectionner l'émulateur dans Android Studio, le résultat est illustré sur la figure (II.12) .

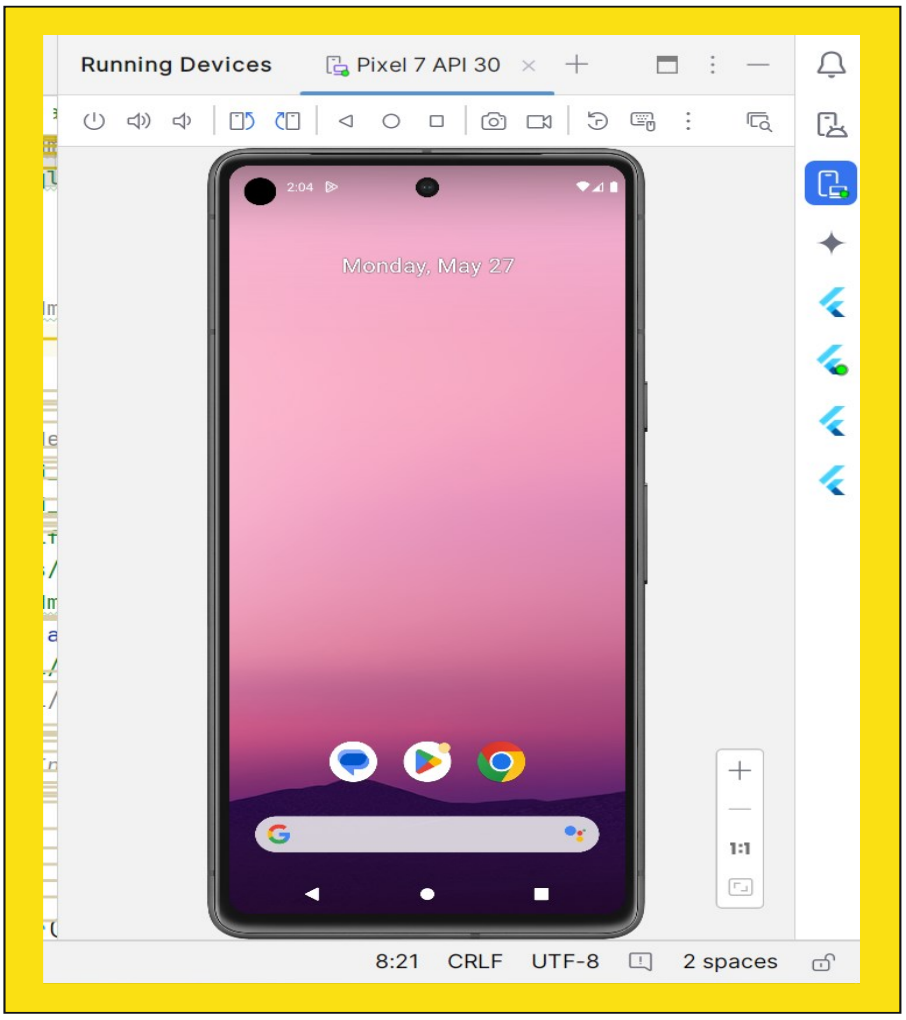


Figure II. 12 :Résultat de l'Emulateur d'Android utilisé .

Ensuite ,on lancer une application virtuelle sur la barre d'outils comme la figure (II.13) :

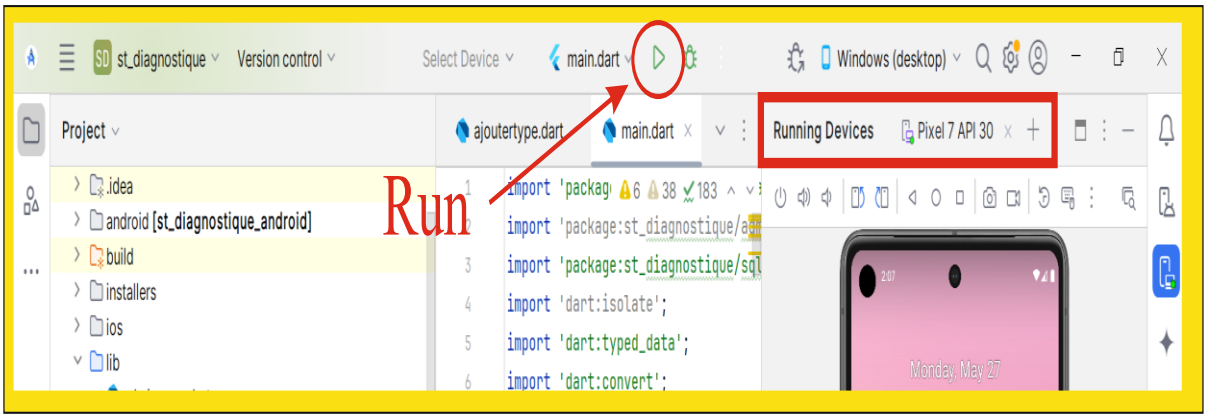


Figure II. 13 : la Barre d'Outils d'Android studio.

Le résultat final de la création de l'application est donné sur la figure (II.14) , a partir de cette étape , notre projet de gestion de l'application est créé .

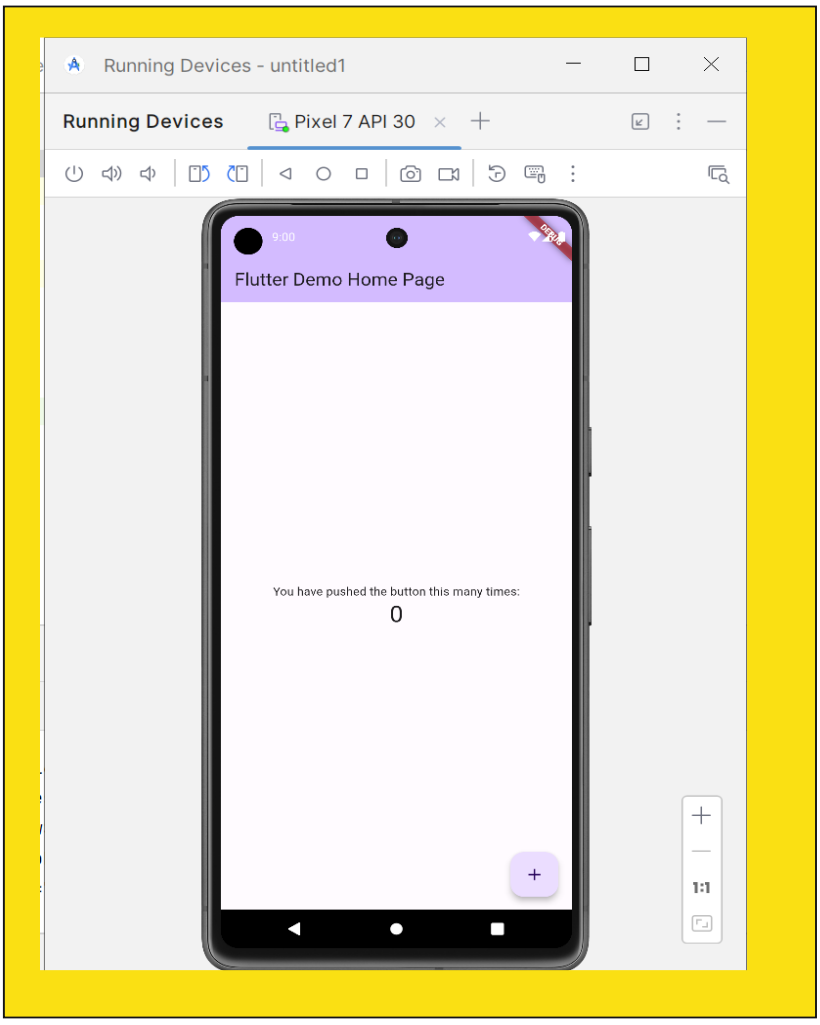


Figure II. 14 : Fenêtre de résultat final obtenue .

II.5 L'organigramme de création de notre application:

Le diagramme montre les étapes du développement d'un projet Flutter Chaque étape représente une phase clé du cycle de développement de l'application, depuis sa création jusqu'à sa finalisation.

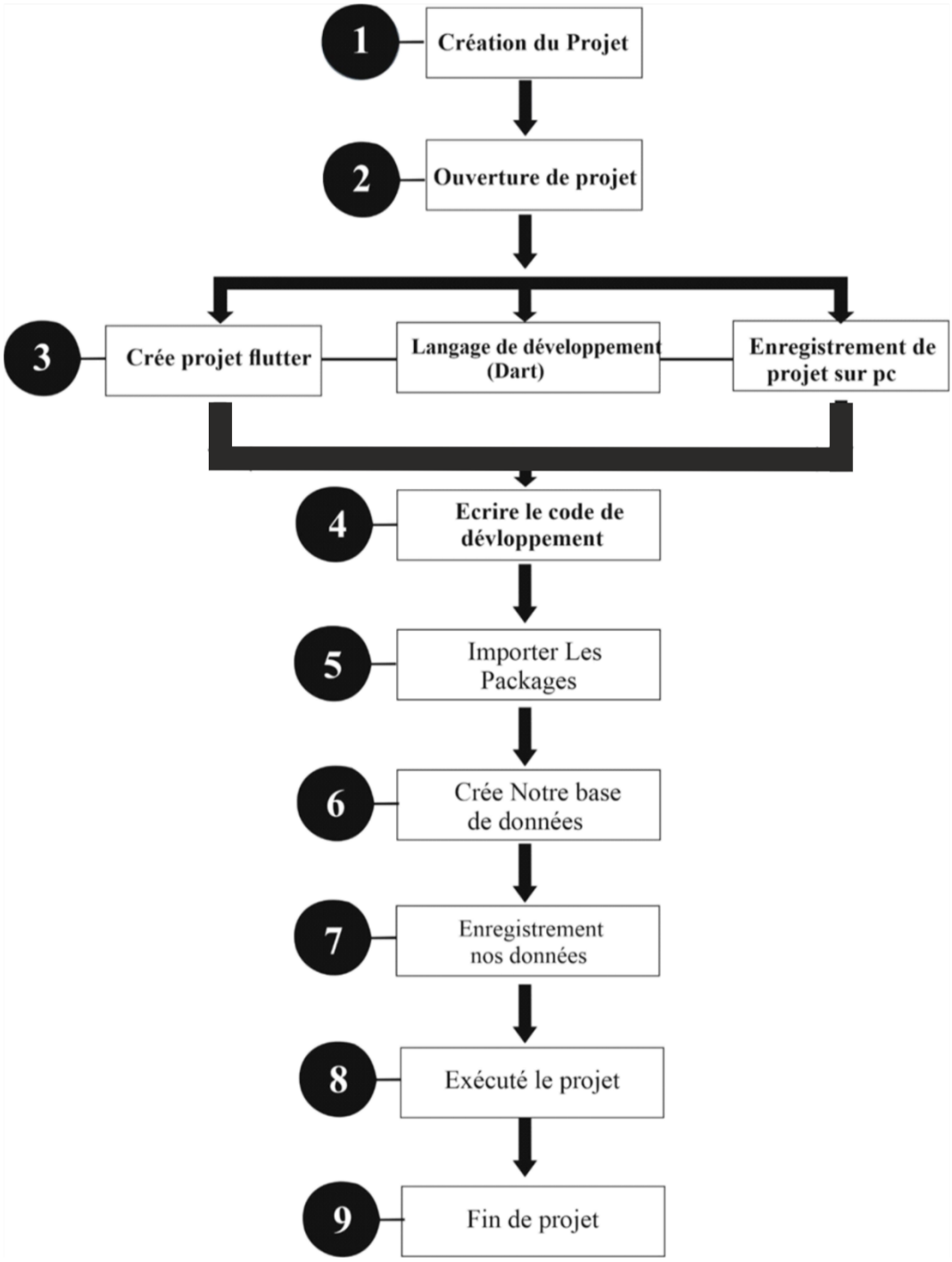


Figure II. 15 : organigramme de création de l'application

II.6 Code de programmation :

Les étapes suivants sont suivies pour la création de l'application :

Etape 1 : Crée des fichiers Dart dans le dossier lib

Une fois notre projet est créé , nous modifions l'application en accédant aux fichiers dans le dossier de notre projet, notamment les fichiers Dart dans le dossier "lib".

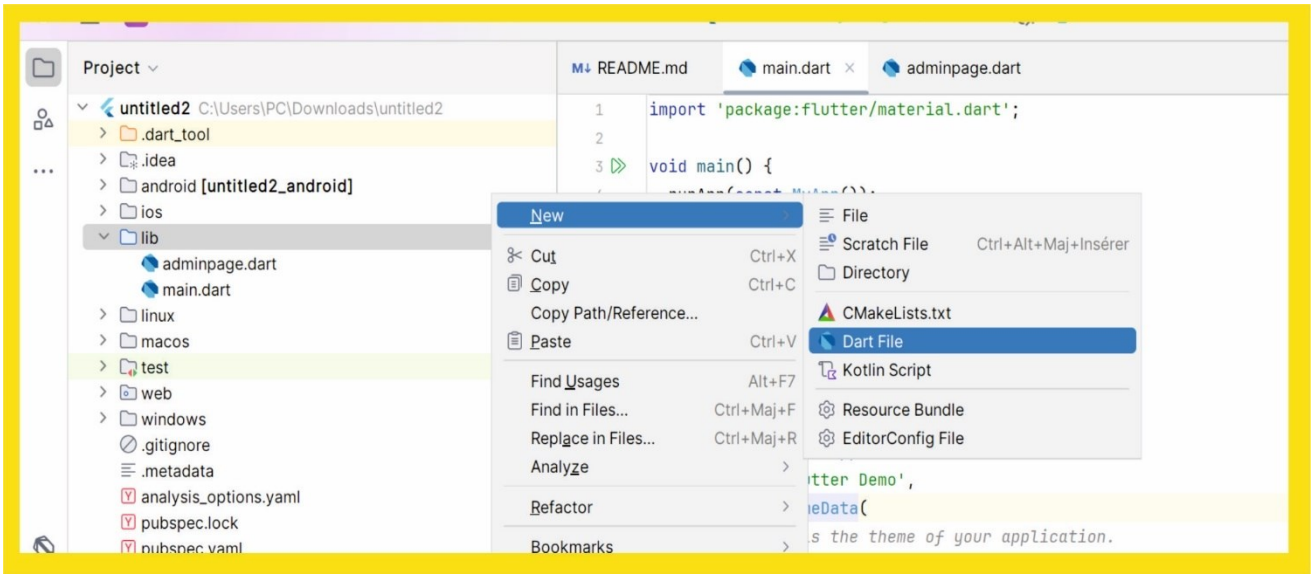


Figure II. 16 : Crée fichier Dart .

Etape 2 : Importer les bibliothèques

Une fois les fichiers Dart créés comme illustré dans la figure (II.17) :

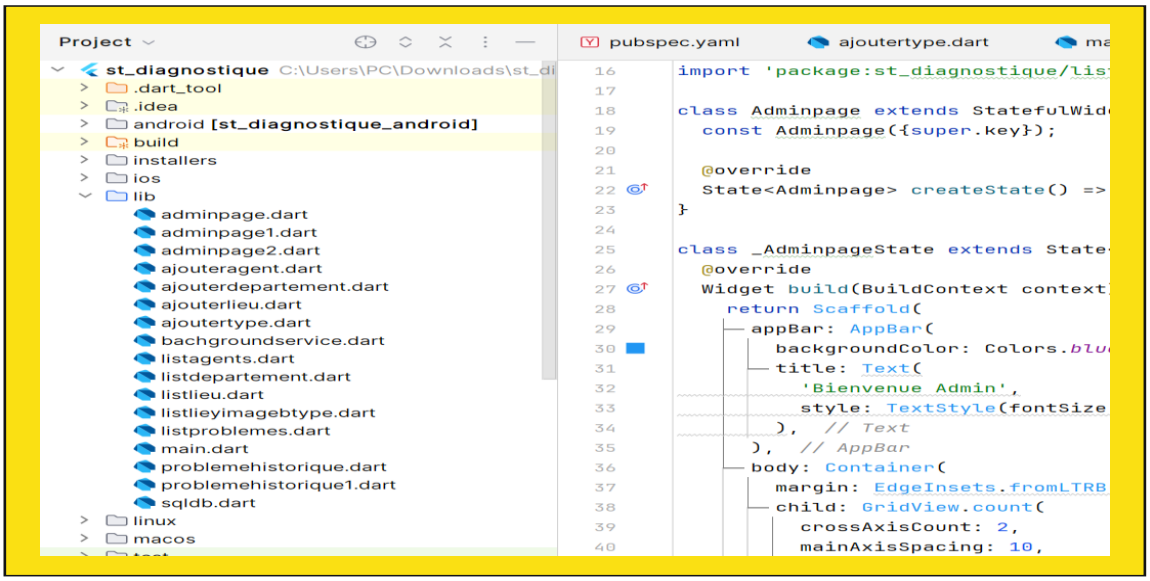


Figure II. 17: Affichage du fichiers lib .

Maintenant, l'ouverture du fichier "pubspec.yaml" pour inclure les nouveaux fichiers Dart que nous avons créés précédemment. Ensuite, l'appuie sur "Pub get" pour mettre à jour les bibliothèques ajoutées à notre programme.

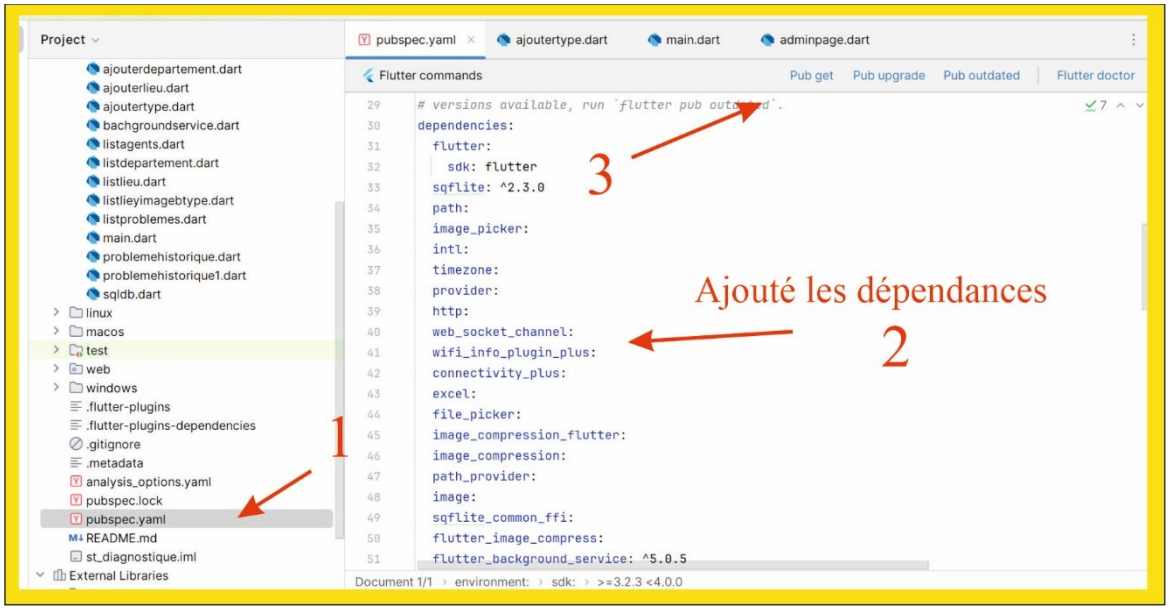


Figure II. 18 : Mettre à jour les bibliothèques .

Nous avons importées les bibliothèques nécessaires à chaque fichiers Dart .

Figure II. 19 : Importation des package .

II.6.1 Les exemples d'activité de notre code :

Dans cette section, nous présenterons un exemple de code pour ajouter un Bouton .

Les code pour ajouter un Botton :

Cette activité permet d'ajouter des boutons comme "Ajouter des agents", "Ajouter des départements" et "Ajouter des lieux".

Tout d'abord, nous créons une classe "AdminPage" pour afficher l'interface de l'application, comme indiqué dans la figure (II.20) :

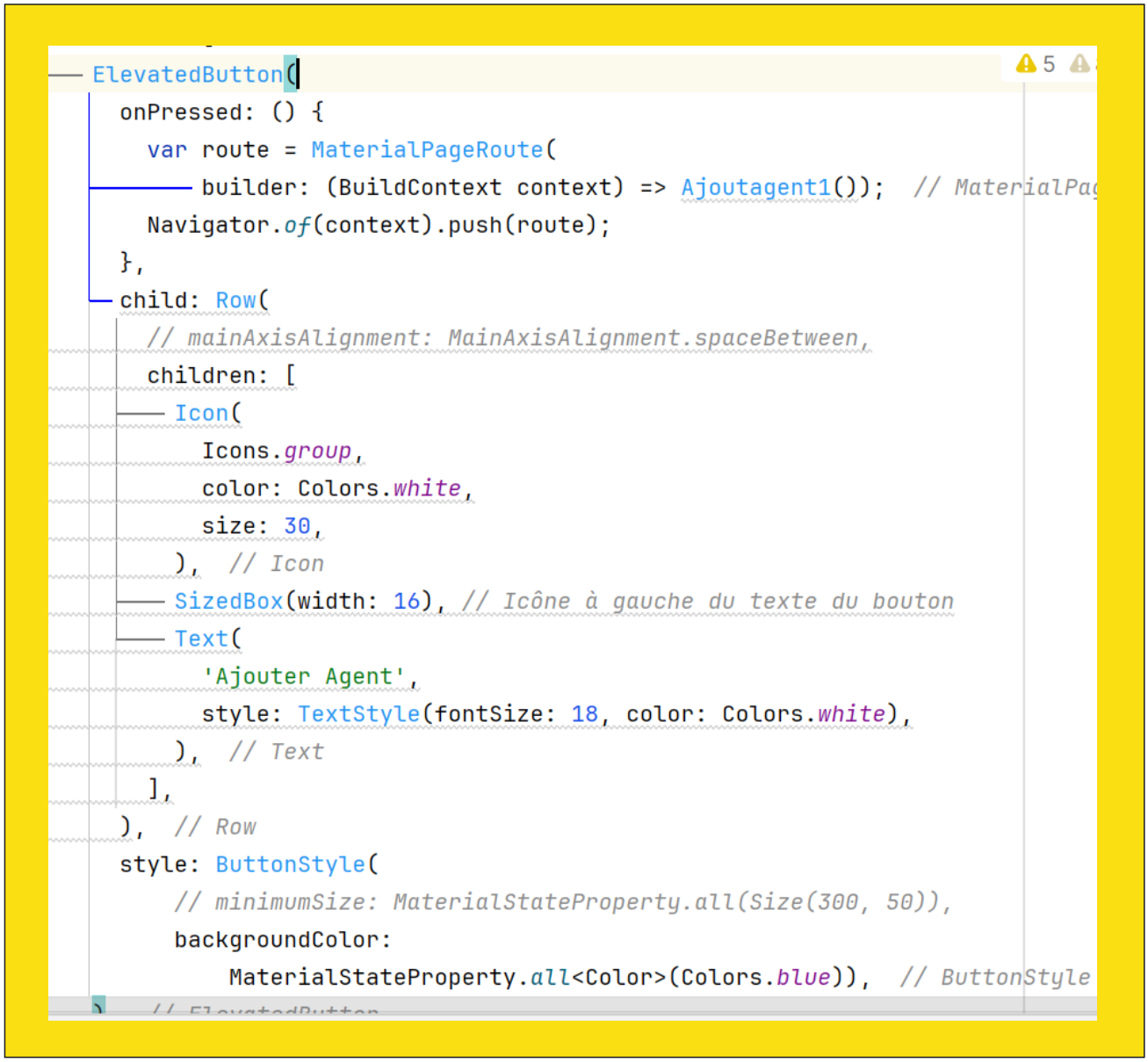
```
class Adminpage extends StatefulWidget {
  const Adminpage({super.key});

  @override
  State<Adminpage> createState() => _AdminpageState();
}

class _AdminpageState extends State<Adminpage> {
  @override
  Widget build(BuildContext context) {
    return Scaffold(
      appBar: AppBar(
        backgroundColor: Colors.blue,
        title: Text(
          'Bienvenue Admin',
          style: TextStyle(fontSize: 30, color: Colors.whi
        ),  // Text
      ),  // AppBar
      body: Container(
        margin: EdgeInsets.fromLTRB(50, 20, 50, 50),
        child: GridView.count(
          crossAxisCount: 2,
          mainAxisSpacing: 10,
          crossAxisSpacing: 10,
```

Figure II. 20 : Class admin page .

Apres nous créons le Bouton **'Ajouter Agent' .**



```
ElevatedButton(
  onPressed: () {
    var route = MaterialPageRoute(
      builder: (BuildContext context) => Ajoutagent1());  // MaterialPag
    Navigator.of(context).push(route);
  },
  child: Row(
    // mainAxisAlignment: MainAxisAlignment.spaceBetween,
    children: [
      Icon(
        Icons.group,
        color: Colors.white,
        size: 30,
      ),  // Icon
      SizedBox(width: 16), // Icône à gauche du texte du bouton
      Text(
        'Ajouter Agent',
        style: TextStyle(fontSize: 18, color: Colors.white),
      ),  // Text
    ],
  ),  // Row
  style: ButtonStyle(
      // minimumSize: MaterialStateProperty.all(Size(300, 50)),
      backgroundColor:
          MaterialStateProperty.all<Color>(Colors.blue)),  // ButtonStyle
  ),  // ElevatedButton
```

Figure II. 21 : Exemple de code pour ajouter un compte agent

La base de données :

Dans cette activité, nous avons utilisé 5 options pour :

- Insérer les données (ajouter agent, département ,type et lieu).
- Insérer les images .
- Lire les données.
- Mettre à jour les données .
- Supprimé les données .

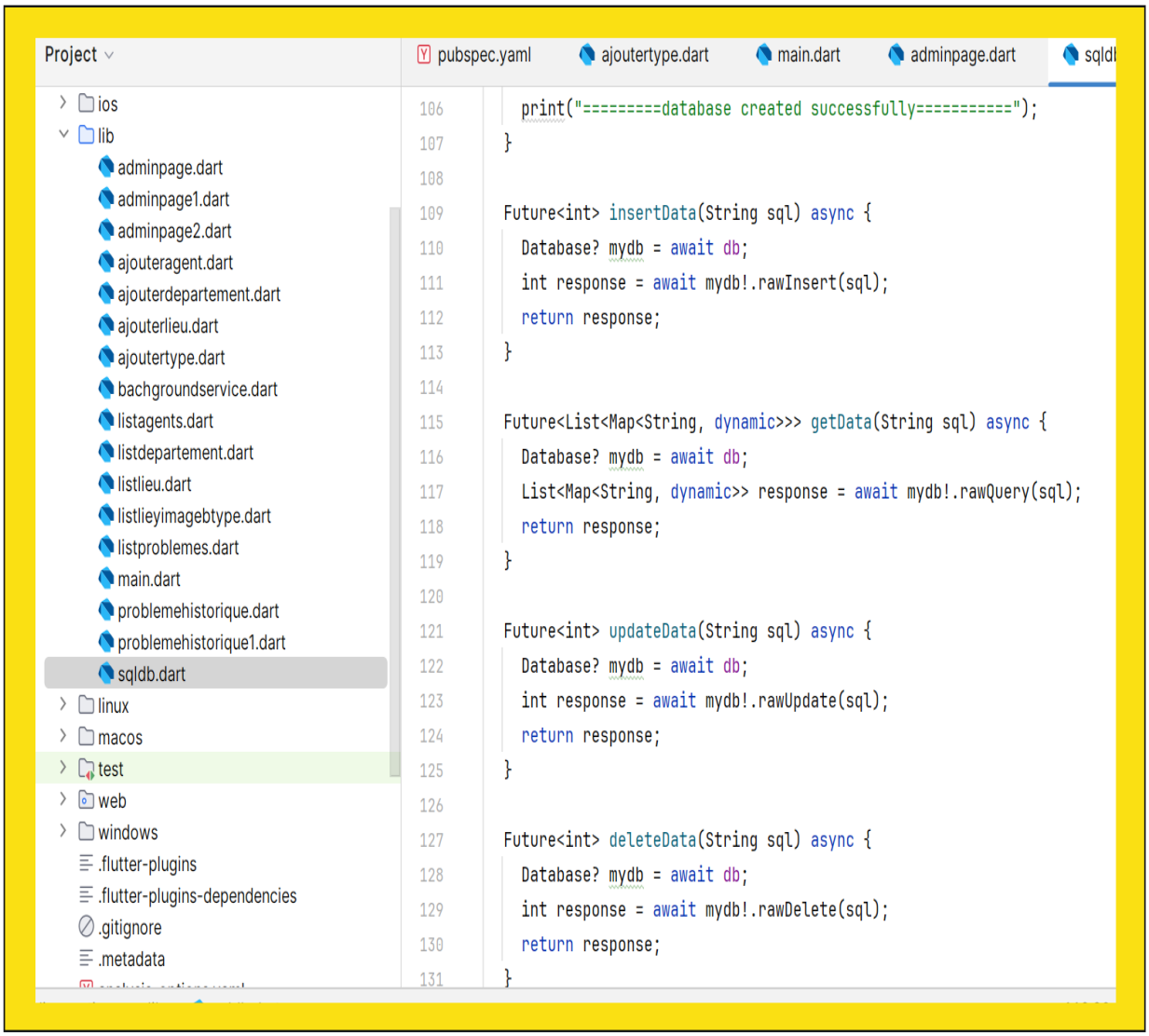


Figure II. 22 : Le code de la base des données.

II.7 Conclusion :

Ce chapitre a permet de présenté les outils et logiciels essentiels pour le développement de notre application, en mettant en avant l'utilisation de Flutter avec Android Studio. Les différentes étapes de création ont été expliquées en détails, le chapitre suivant sera dédié à la discussion des résultats des tests opérés et les résultats obtenus ont été présentés de manière argumentée.

Chapitre III :

Développement de l’application et résultats obtenus

Chapitre III : Développement et résultats obtenus de l'application

III.1 Introduction :

Dans ce chapitre, nous présenterons les résultats obtenus après le développement de l'application proposée. Ces résultats incluent les fonctionnalités clés de l'application, telles que le signalement des problèmes en temps réel, la gestion de l'historique des problèmes, la communication directe entre les agents et le superviseur, et l'exportation des données vers un fichier Excel. Nous examinerons également les performances de l'application en termes de fiabilité de la connexion et de capacité de gestion des utilisateurs. De plus, nous discuterons de l'interface utilisateur, tant pour les agents que pour le superviseur. Enfin, une discussion approfondie sera menée pour évaluer les points forts, les limitations et les perspectives d'amélioration de l'application, afin de proposer des solutions et des améliorations futures pour optimiser son utilisation et sa performance.

III.2 Développement de l'application :

III.2.1 Les interfaces crées :

Nous avons développé une application proposant deux interfaces distinctes : une pour les ordinateurs et l'autre pour les smartphones .

a. Interface application Embaquée :

Cette application (st_ maintenance) a été développée pour être utilisé via les téléphones, elle sera gérée par le superviseur pour envoyer et recever des informations, des agents chargés de signaler les problèmes rencontré à la faculté.

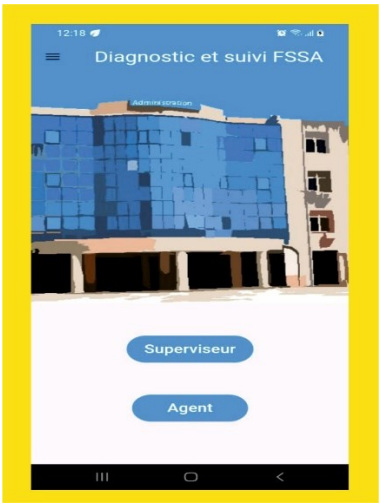


Figure III. 1 : Interface de l'application Embarqués sur un smartphone.

b. Interface application PC :

Le service de maintenance de la faculté est composé d'un responsable de service (superviseur) et d'agents polyvalents de maintenance (Agent dans l'application).Ces agents sont rattachés fonctionnellement à ce service .Le rôle du superviseur est de crée, et de supprimer les comptes d'interfaces de ses agents.

L'application (st-diagnostic) est destinée à être utiliser par le superviseur qui recevra les problèmes signalés par l'agent via l'application mobile et aura la capacité d'ajouter et de supprimer des comptes agents, des lieux et des problèmes. De plus, il pourra exporter l'historique des problèmes signalés .

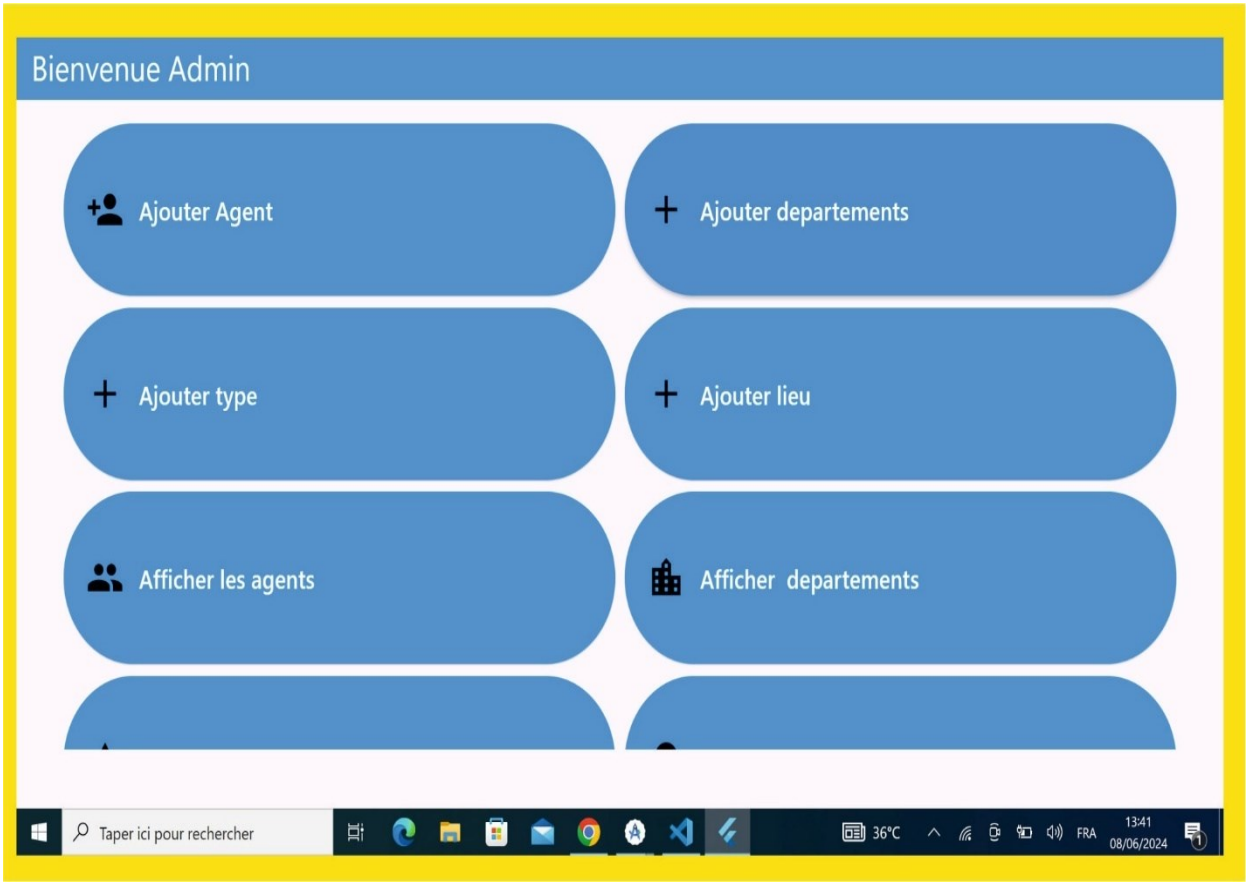


Figure III. 2 : Interface application PC.

III.4 Organigramme d'utilisation de l'application:

Le Organigramme d'utilisation ci-dessous (Figure III.3) expose les diverses fonctionnalités de l'application, en suivant les options proposées pour identifier, analyser et diagnostiquer les problèmes existants.

III.4.1 Interface Embarquée (St_ maintenance) :

Cet organigramme montre clairement les rôles et responsabilités distincts du superviseur et de l'agent au sein de l'application, ainsi que les fonctionnalités accessibles à chacun après connexion.

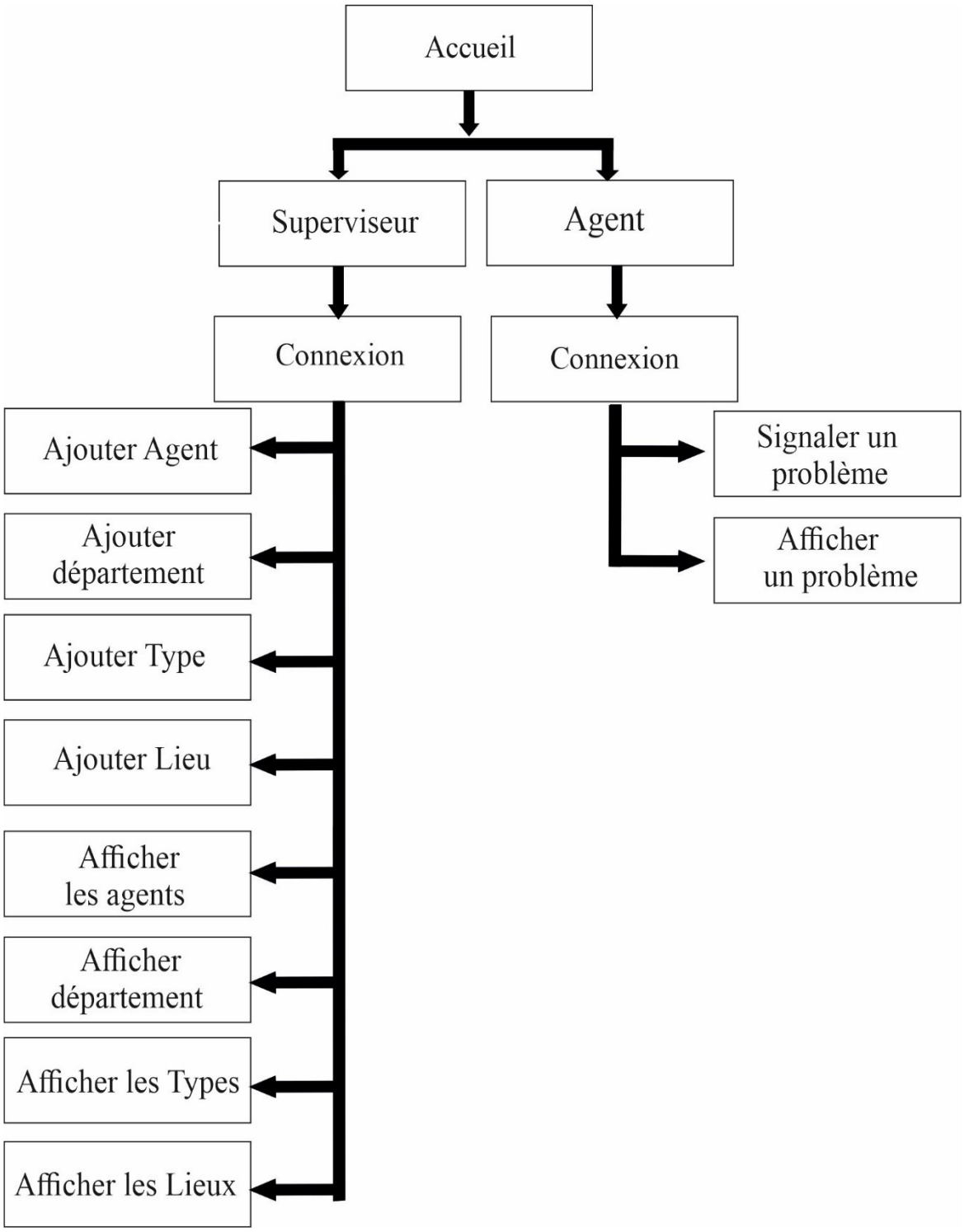


Figure III. 3 : Organigramme de l'utilisation de l'application par le Superviseur et l'agent.

III.4.2 Interface PC (St_ Diagnostic) :

Cet organigramme détaille les actions que le superviseur peut effectuer depuis la page d'accueil de l'application, offrant une vue claire sur la gestion des agents, départements, types de problèmes, lieux, ainsi que sur le suivi et l'affichage des problèmes signalés et leur historique.

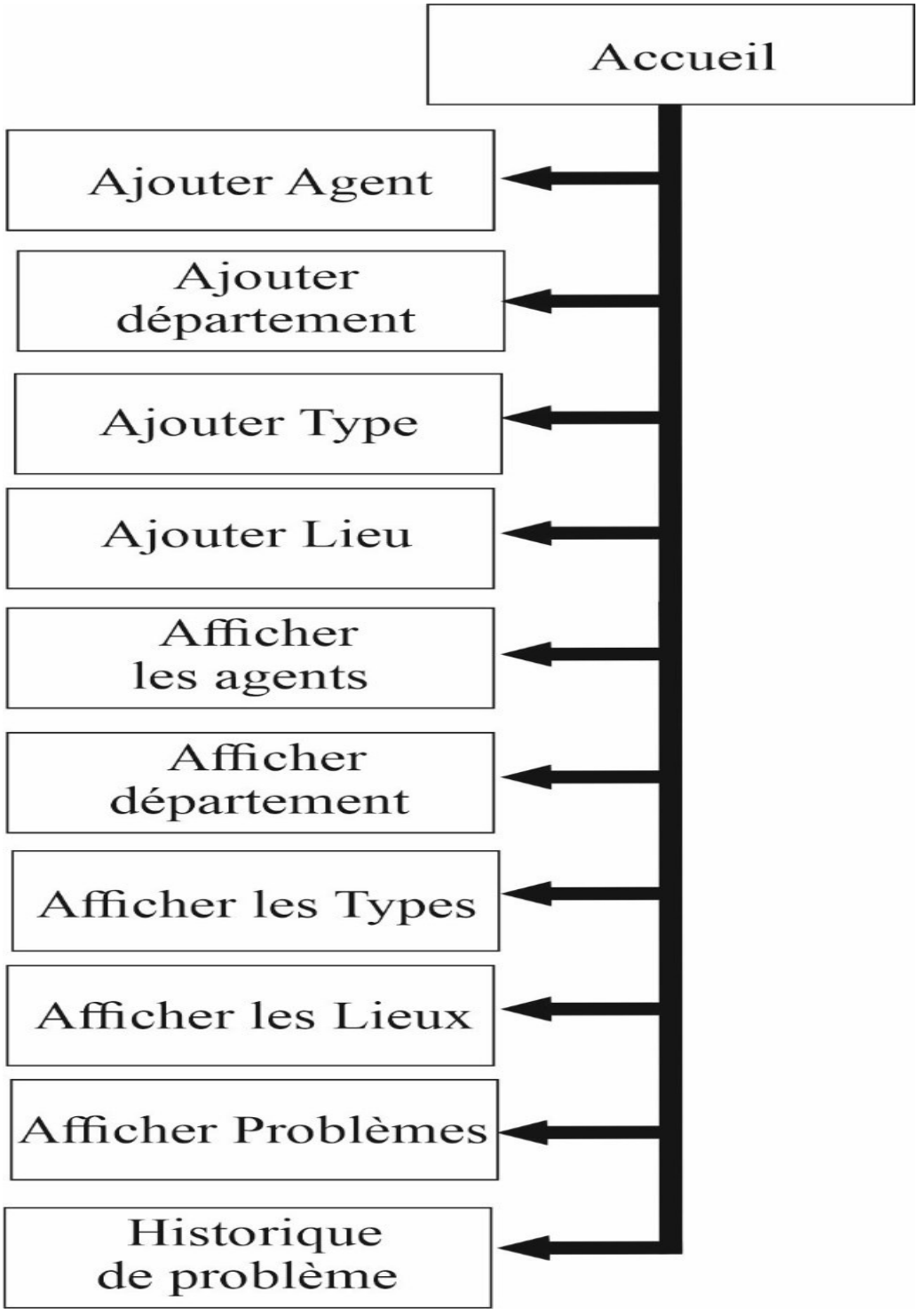


Figure III. 4 : Organigramme de l'utilisation de l'application par le Superviseur

.

III.5 Les activités de l'application :

Dans ce qui suit, nous présenterons le fonctionnement de l'application, en identifiant les agents et le superviseur, et en affichant les interfaces de chaque opération effectuée par l'agent et le superviseur.

III.5.1 Le superviseur :

A. Définition : Le superviseur est une personne qui gère l'application, ayant la liberté de l'utiliser et le pouvoir d'ajouter ou de supprimer des comptes agents, départements et lieux .

B. Rôle dans l'application : le superviseur a plusieurs rôles dans le fonctionnement de l'application tels que : :

- Ajouter ou supprimer les comptes des agents
- Ajouter les départements, lieux ,types
- Solutionner les problèmes signalé par les agents
- Suivi les travaux engagés .
- Diagnostic les maintenances opérées .

C. Les paramètres :

Nom et le prénom : **m (il représente le nom et prénom de superviseur)**

Le Mots de Passe : **1234**

L'accès : **Le superviseur a l'accès au deux interfaces de notre application (interface PC et interface Embarquée) .**

D. Des captures d'écrans sur les applications gérer par le superviseur :

Interface Embarquée :

Etape 1 : Le superviseur saisit ses données, et ajoute un compte d'un agent.

Figure III. 5 : Interface d'ajout un compte d'agent par le superviseur

.

Etape 2 : Nous avons ajouté les données de l'agent et importé sa photo faciale. Ensuite, l'agent apparaîtra dans la liste des agents, comme illustré sur la figure (III.6) :

Figure III. 6 : Résultats d'ajout un compte d'agent par le superviseur .

Interface PC :

Figure III. 7 : Mode d'emploi d'un ajout d'un compte agent par le superviseur .

III.5.2 Les Agents :

A. Définition : Un agent est une personne chargée de surveiller le milieu de notre faculté. Il a pour responsabilité de signaler les problèmes qui se produisent et de transmettre les informations pertinentes au superviseur.

B. Rôle dans l'application : Dans notre application, l'agent a pour seul rôle de signaler les problèmes qui se produisent sur le terrain de la faculté .

C. Les paramètres :

Le nom et prénom :**MOUSSA**

La Matricule : **BOUKHELF**

Identité (image faciale) :

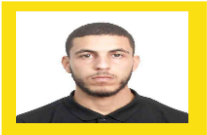


Figure III. 8 : Exemple d'image faciale de l'agent.

L'accès : **Signaler les problèmes**

Utilisation : **Interface Embarquée .**

D. Nombre des agents : Dans notre projet, nous avons prévu d'utiliser initialement 3, mais nous avons la possibilité d'en ajouter jusqu'à 7 selon le besoin.

Remarque : Lorsque un agent signale un problème qui a déjà été rapporté par un autre collègue, un message d'erreur s'affiche pour éviter les redondances et assurer une meilleure organisation.

E. Des captures d'écrans sur l'applications gérer par l'agent :

Etape 1 **:** L'agent doit saisir son nom et la matricule pour accéder à l'application.

Figure III. 9 : Saisie des données relatives à l'agent.

Etape 2**:** Nous examinerons à cette étape comment l'agent signalera un problème et comment celui-ci sera affiché au niveau du superviseur .

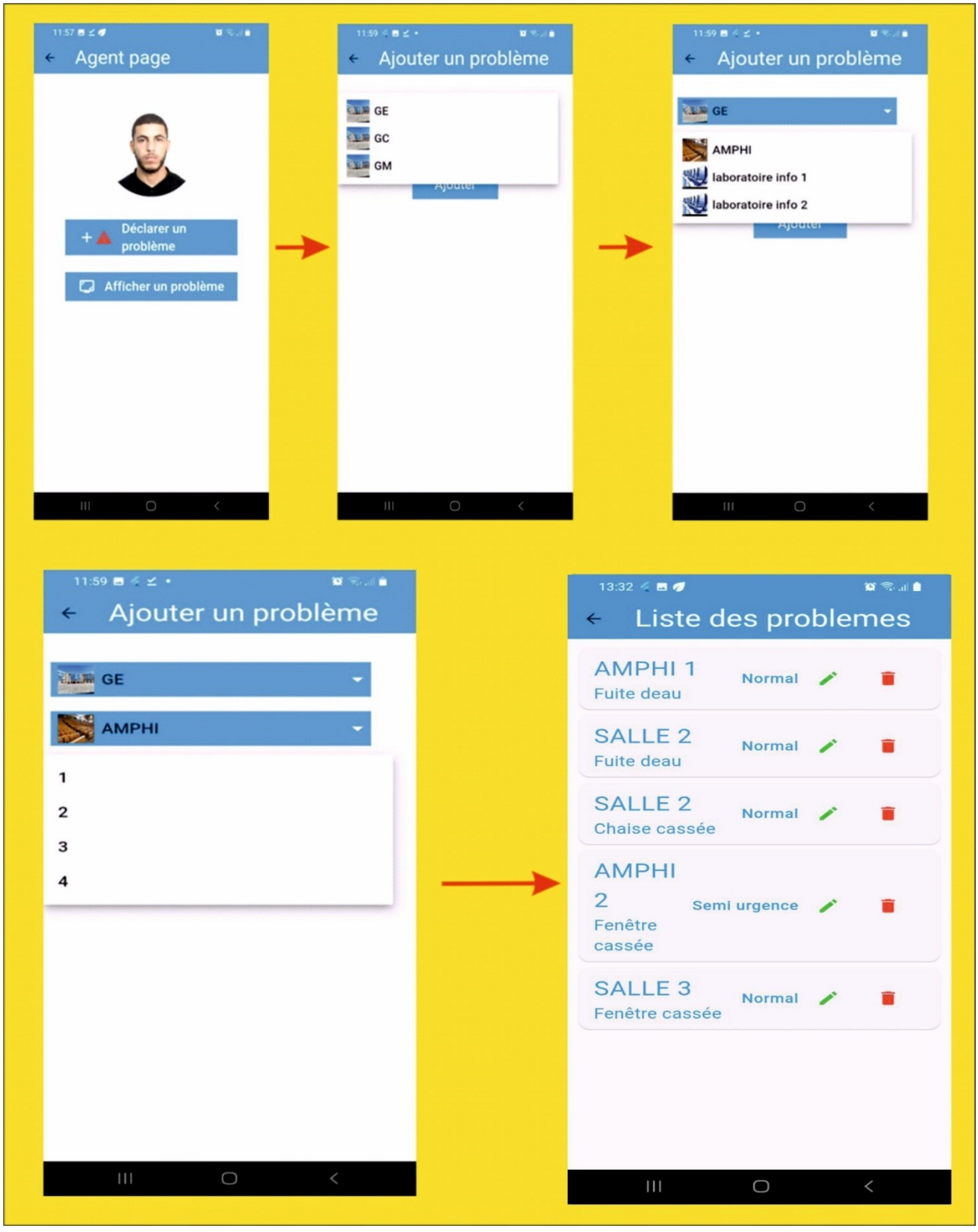


Figure III. 10 : Procédure de signalement d'un problème par un agent.

III.5.3 Des captures d'écran générale sur l'application :

A. L'interface Pc :

La figure suivante montre l'interface utilisateur de l'application par le superviseur .

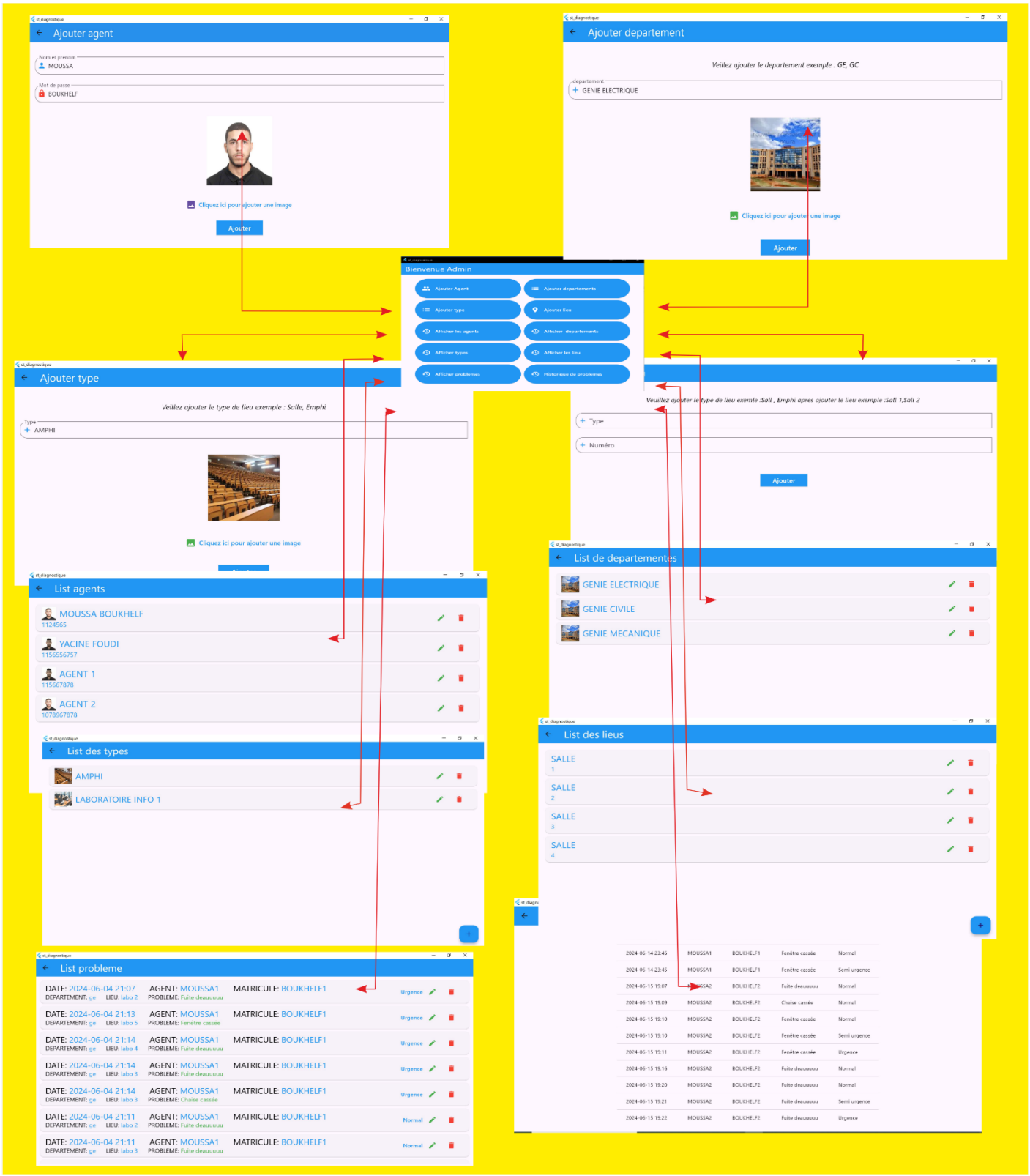


Figure III. 11 : Interface d'accueil relative au superviseur .

B. L’interface de L’agent :

La figure suivante montre l'interface utilisateur de l'application par l’agent .

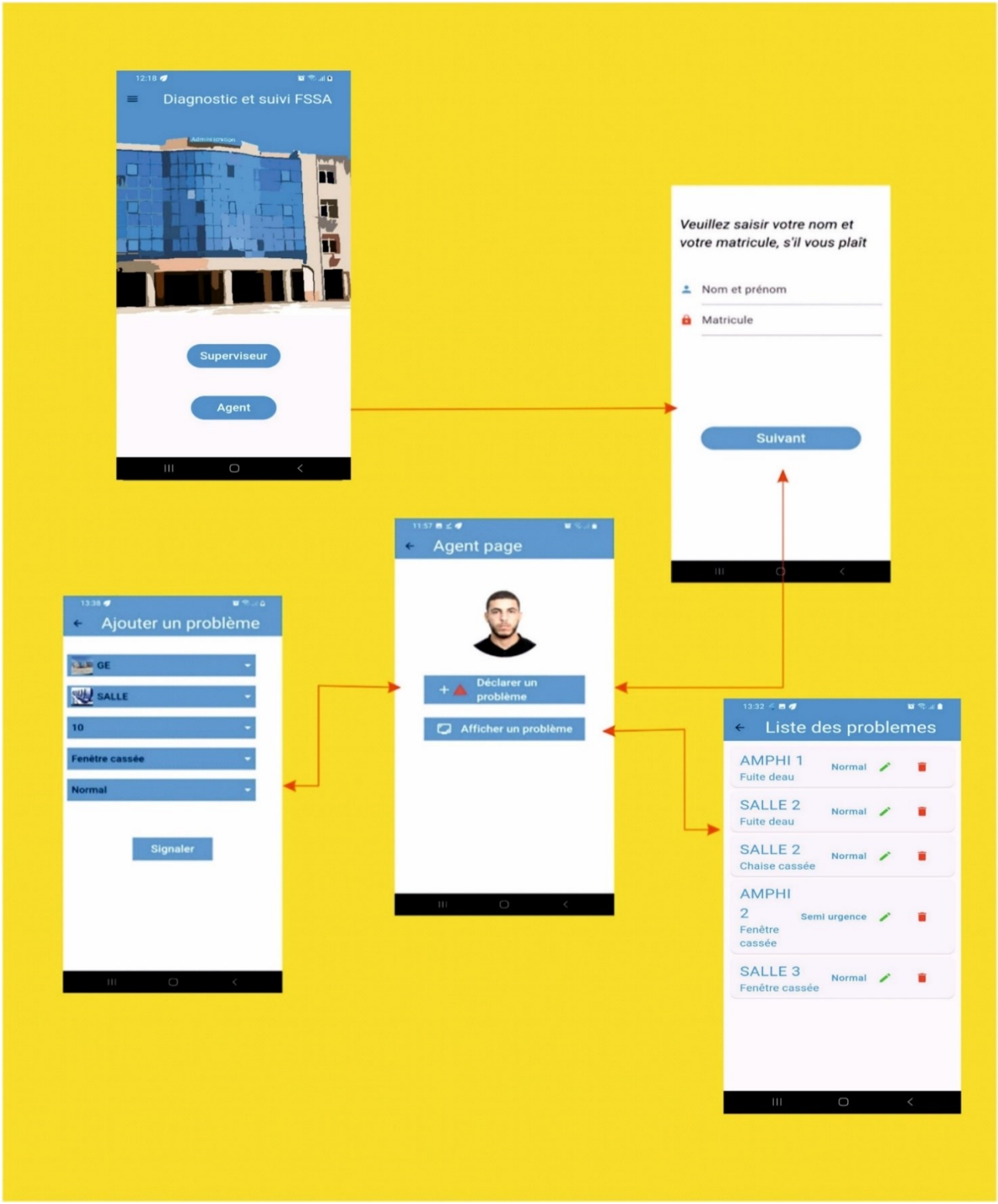


Figure III. 12 : Interface d’accueil relative à l’agent**.**

III.6 Les résultats obtenus de notre Application :

Dans cette section, nous exposerons les résultats des tests effectués pour la validation de l'application.

Résultats de Test 1 :

Dans ce test, l'agent détectera et signalera une anomalie via notre application. Nous allons détailler le processus de détection et de signalement, en précisant les étapes techniques impliquées.

Explication : Lorsqu'un agent signale un problème au serveur en incluant les détails du problème, le serveur réceptionne ces informations et envoie un message de confirmation à l'agent. Ce message est présenté sous forme de notification pour l'informer que le problème a été pris en compte .

Des captures d'écran de test 1 :

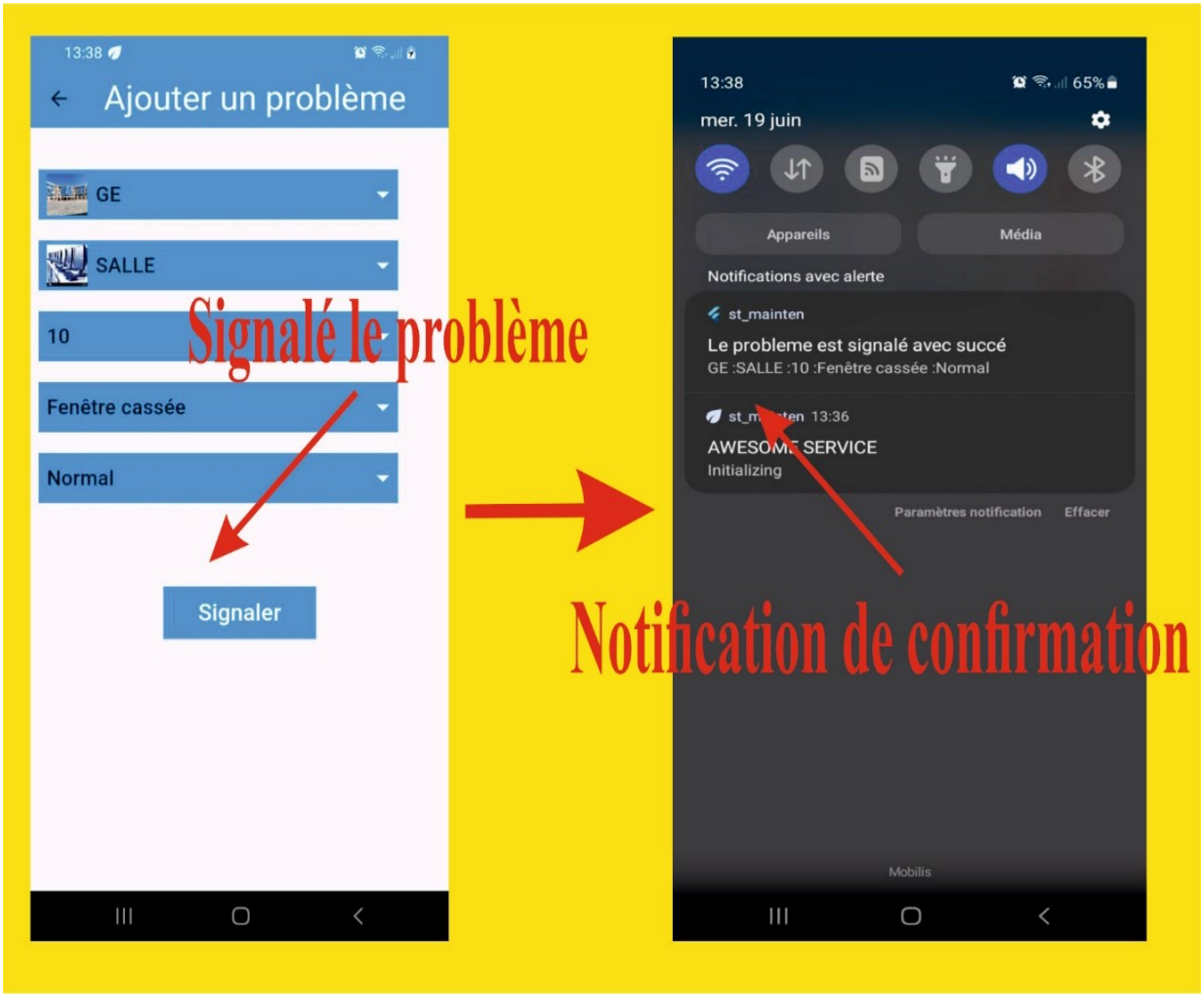


Figure III. 13 :Notification de confirmation d'un problème signalé par un agent .

Résultats de Test 2 :

Dans ce test , le superviseur rajout un compte d'agent

Explication : Lorsqu'un superviseur souhaite ajouter un nouvel compte pour l'agent, il se connecte à l'application à partir d'un PC ou d'un téléphone portable. Ensuite, il appuie sur le bouton "Ajouter agent", saisit le nom, la matricule et la photo d'identité de l'agent, puis appuie sur le bouton "Ajouter". Un message s'affiche alors, indiquant qu'il faut attendre un moment pendant que l'opération se termine.

Des captures d'écran de test 2 :

A. Interface Embarquée :

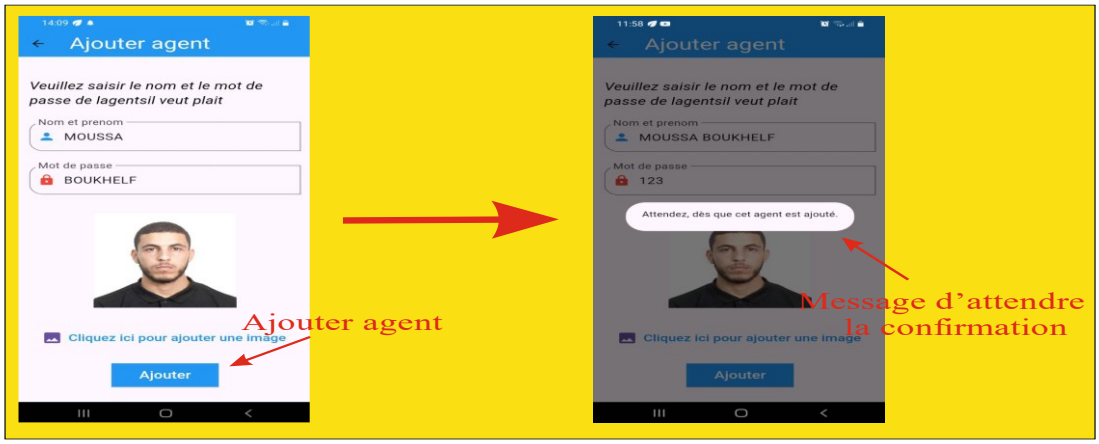


Figure III. 14 : Message d'attendre la confirmation (Embarquée)

B. Interface pc :

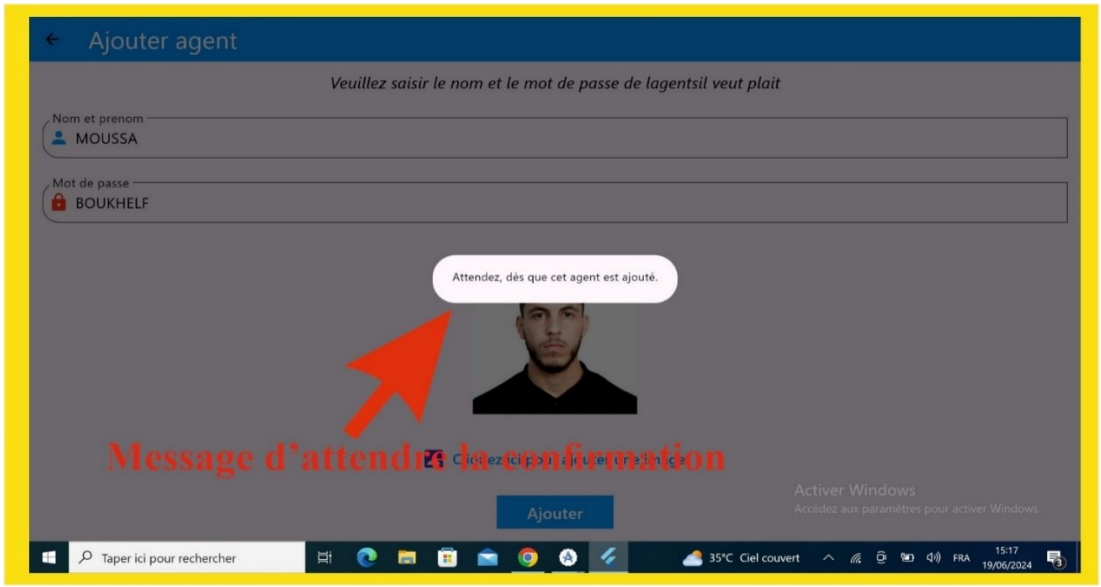


Figure III. 15 : Message d'attendre la confirmation (pc).

Résultats de Test 3 :

Dans ce test nous avons évalué la capacité d'accueil de notre application en termes de nombre d'utilisateurs.

Explication : Notre application en développement est actuellement configurée pour être utilisée par un total de huit utilisateurs, dont un superviseur et sept agents. Cependant, ce nombre n'est pas définitif car l'application est toujours en phase de développement. Nous avons pu atteindre cette limite en utilisant un modem pour la connexion entre l'application et le serveur, avec une capacité maximale de huit téléphones portables connectés au modem, donc avec une contrainte technologique .

Schéma :

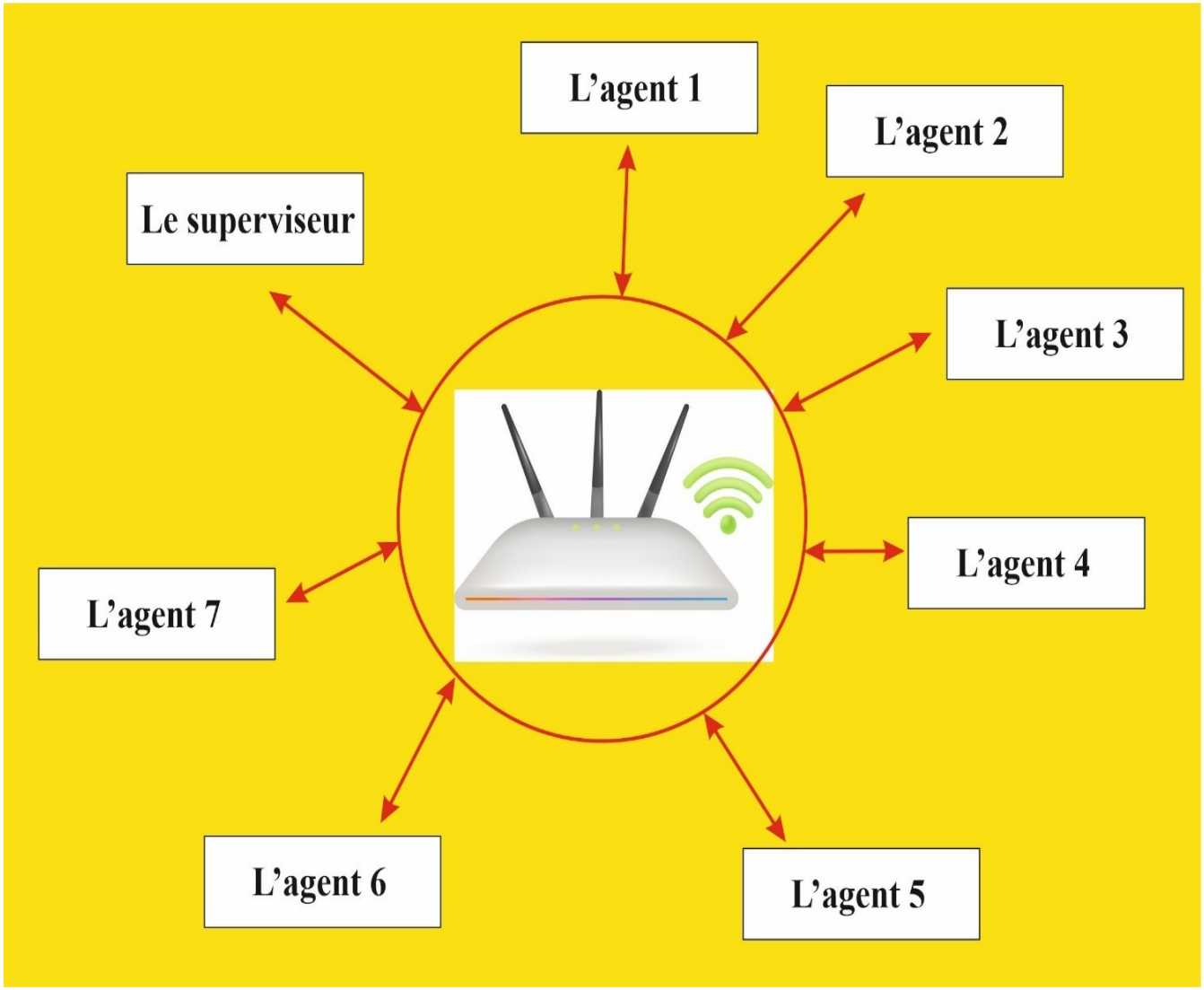


Figure III. 16 : Schéma de nombre d'utilisateurs .

Résultats de Test 4 :

Nous allons tester la distance Approximative du signal pour assurer une utilisation efficace de l'application.

Explication : Notre application utilise un réseau Wi-Fi basé sur un modem. Nous avons déterminé que la distance approximatif à laquelle un téléphone portable peut être connecté au modem tout en assurant une transmission fiable des informations D'après le résultat de la figure (III.17) , la distance est d'environ 30 mètres

Le résultat de la figure (III.17) est obtenu à l'aide de nos modeste moyens matériels , plus particulièrement notre modem personnel .

Ce pendant , ce résultat peut être facilement amélioré en utilisant le réseau wifi de la faculté , l'optimisation de cette distance de communication sera notamment plus grande en utilisant l'internet. Ce moyen de communication n'a pas été testé dans le cadre de ce PFE

Graphe de distance :

Figure 0.I

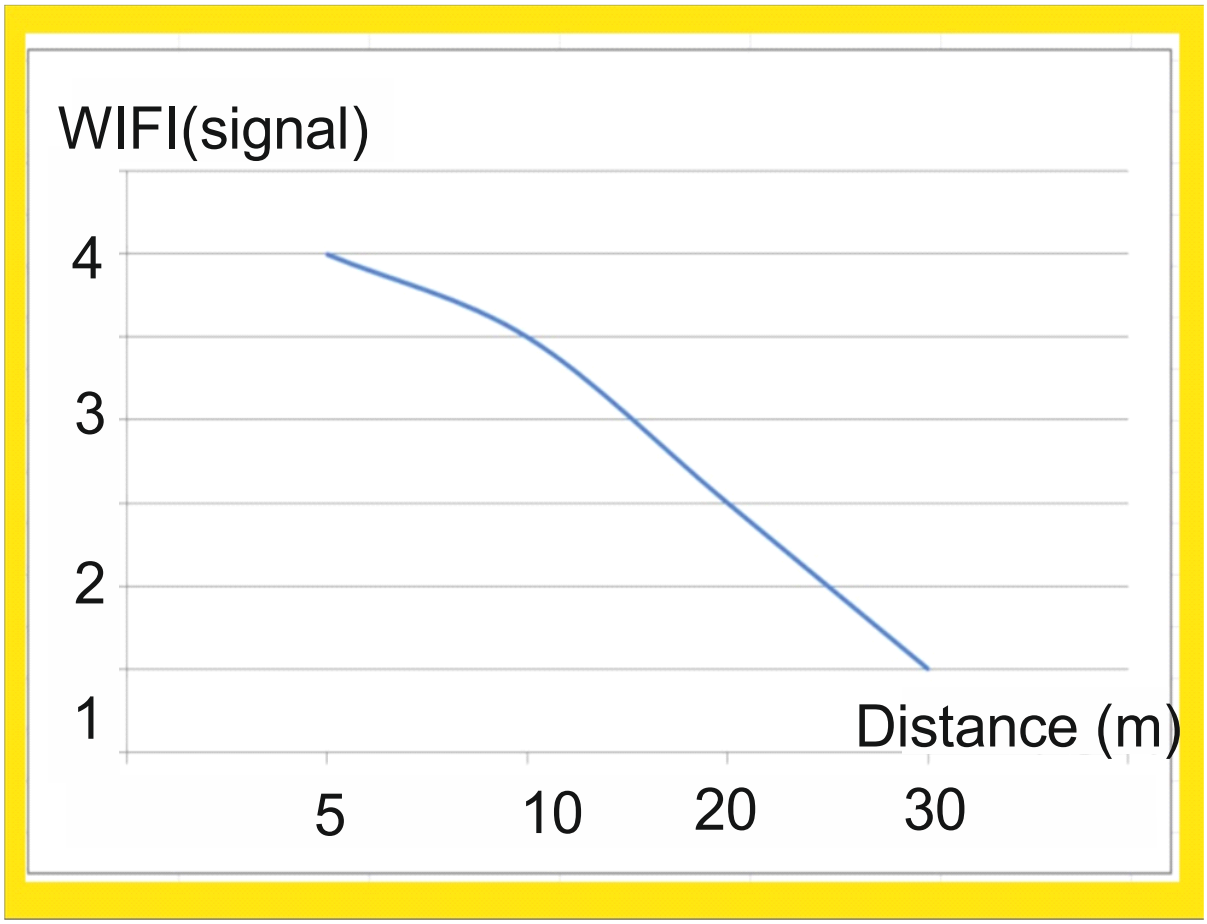


Figure III. 17 : Graphe de distance .

Résultats de Test 5 :

Dans ce test, nous allons afficher les problèmes signalés sur une période d'une semaine.

Explication : Nous allons observer comment l'agent signale des problèmes sur une semaine et comment ces signalements sont enregistrés dans l'historique durée de l'application, Ensuite, nous allons exporter cet historique vers des fichiers Excel.

Des captures d'écran de test 5 :

Figure III. 18 : Les problèmes signalés pendant une semaine .

Résultats de Test 6 :

Dans ce test, le superviseur supprime un compte d'un agent supposé démissionnaire .

Explication : Le superviseur supprime le compte l'agent de l'application parce qu'il a démissionné de son poste, ce qui signifie que le service de maintenance n'a plus besoin de ses services. De plus, cela évite les anomalies d'utilisation de notre application, car elle est spécifiquement destinée au service de maintenance de notre faculté.

Résultats de Test 7 :

L'ajout d'images à notre application est un atout important pour notre projet.

Explication : Lorsque le superviseur souhaite ajouter un agent il va entrer le nom et le matricule de l'agent, puis importer l'image de l'agent à partir du téléphone portable et de l'ordinateur pour mieux l'identifier. C'est une approche pratique pour garantir une identification précise de chaque agent.

Remarque : Le superviseur a la possibilité d'importer les images de l'agent, du département et des lieux.

Capture d'écran :

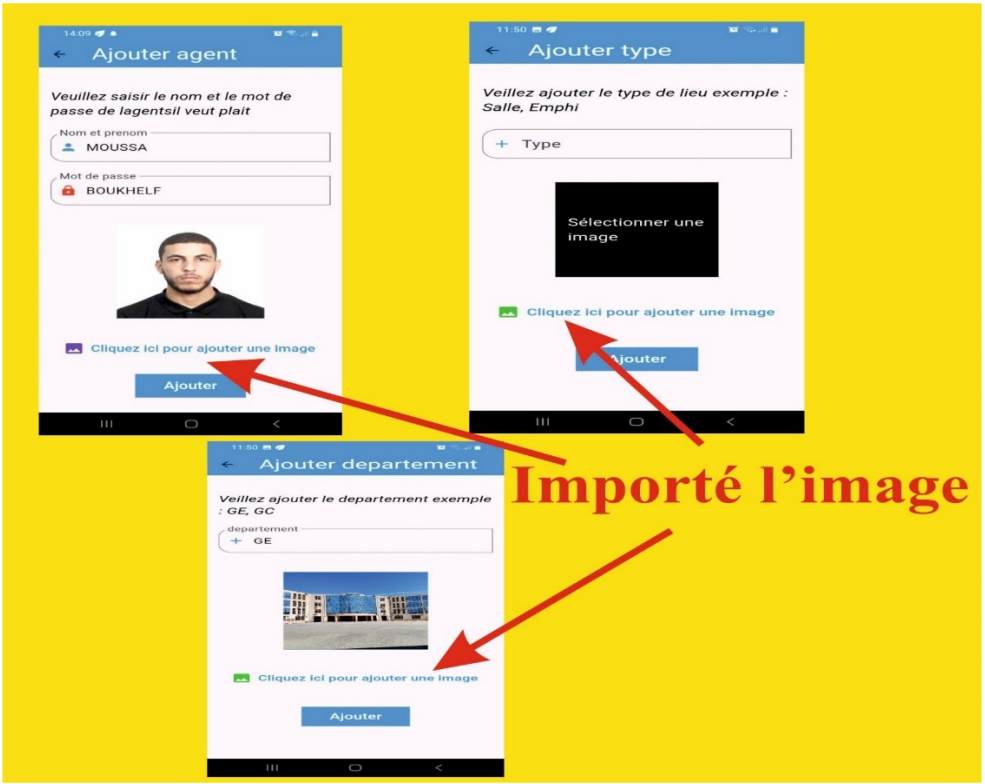


Figure III. 19 : Exemple image importée d' un agent

Résultats de Test 8 :

Lorsque un agent signale un problème, celui-ci s'affiche dans la liste des problèmes signalés.

Explication : Dans notre projet, nous avons mis un accent particulier sur la précision des détails des problèmes signalés par les agents. Lorsqu'un problème est signalé par un agent, il est ajouté à la liste des problèmes avec des informations détaillées incluant la date et l'heure de signalement, le département concerné, le lieu, ainsi que la matricule et le nom de l'agent ayant effectué le signalement.

Capture d'écran :

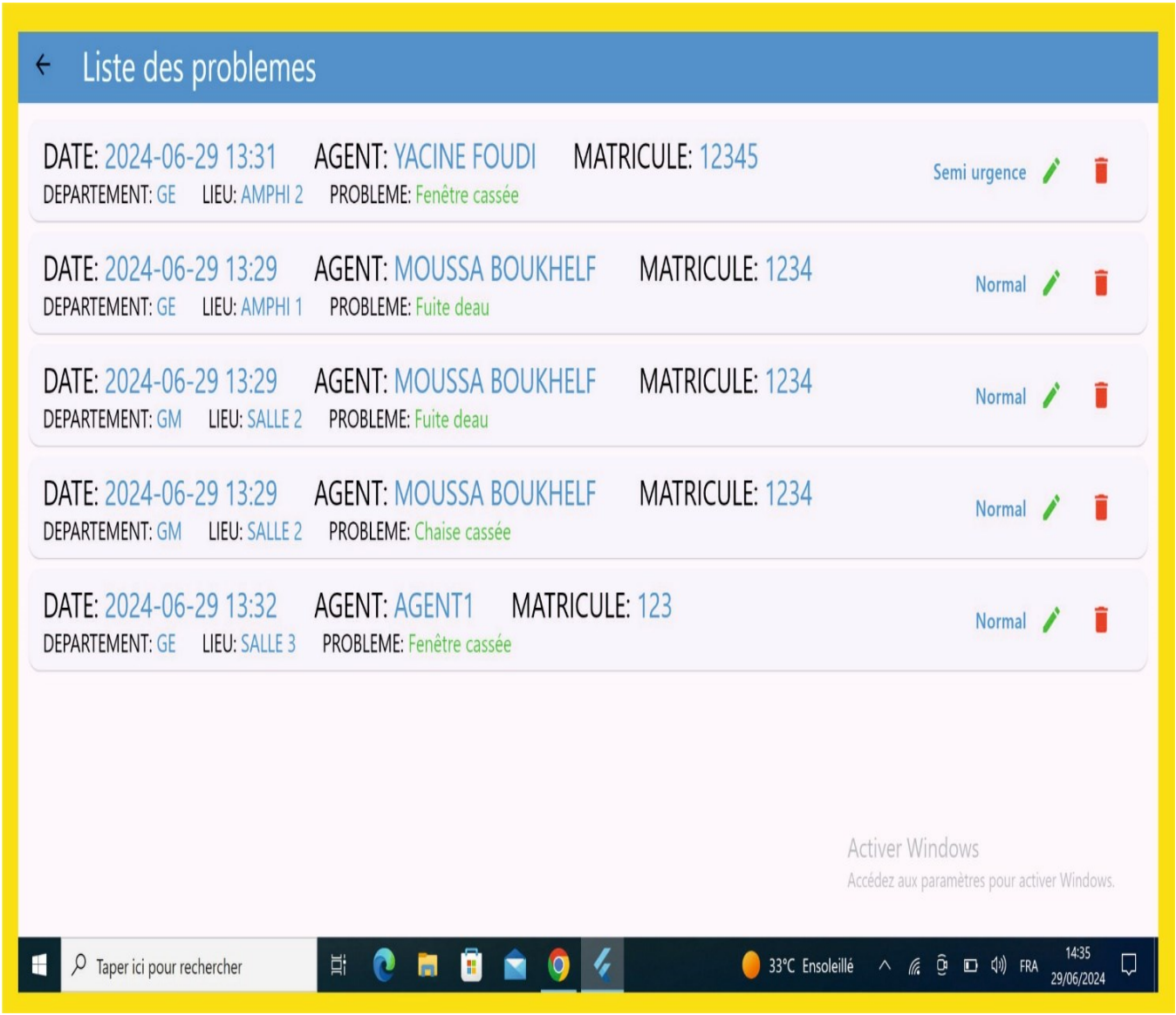


Figure III. 20 : Exemple de Liste des problèmes signalés .

Résultats de Test 9 :

Exporte l'historique est une nouvelle fonctionnalité qui permet d'exporter l'historique des problèmes signalés vers un fichier Excel.

Explication : Dans notre application, un historique est maintenu pour les problèmes signalés, comprenant la date et l'heure de signalement, le nom et la matricule de l'agent concerné, ainsi que l'état actuel du problème. Le superviseur a la capacité d'exporter cet historique vers un fichier Excel pour une gestion plus détaillée.

Capture d'écran :

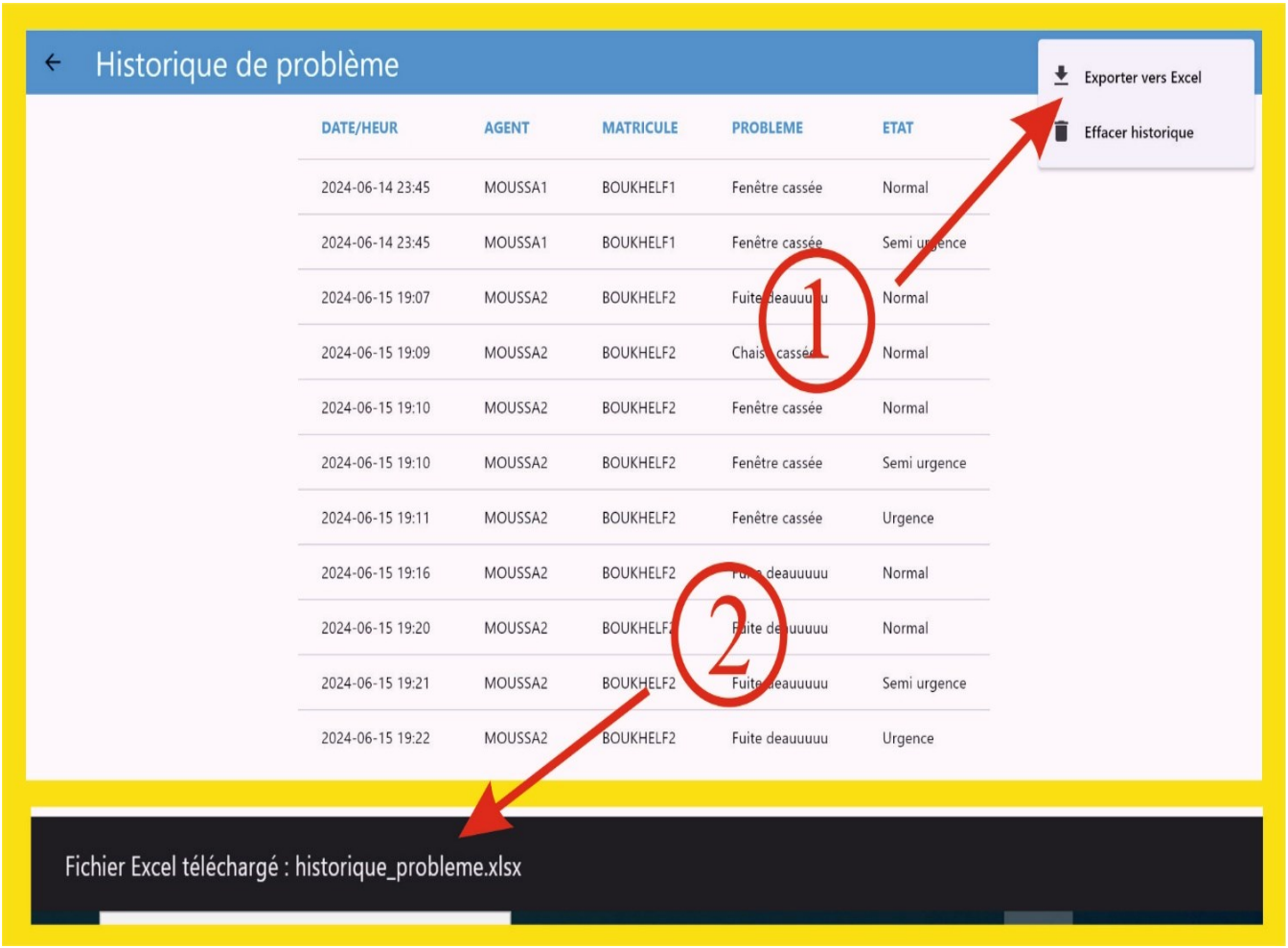
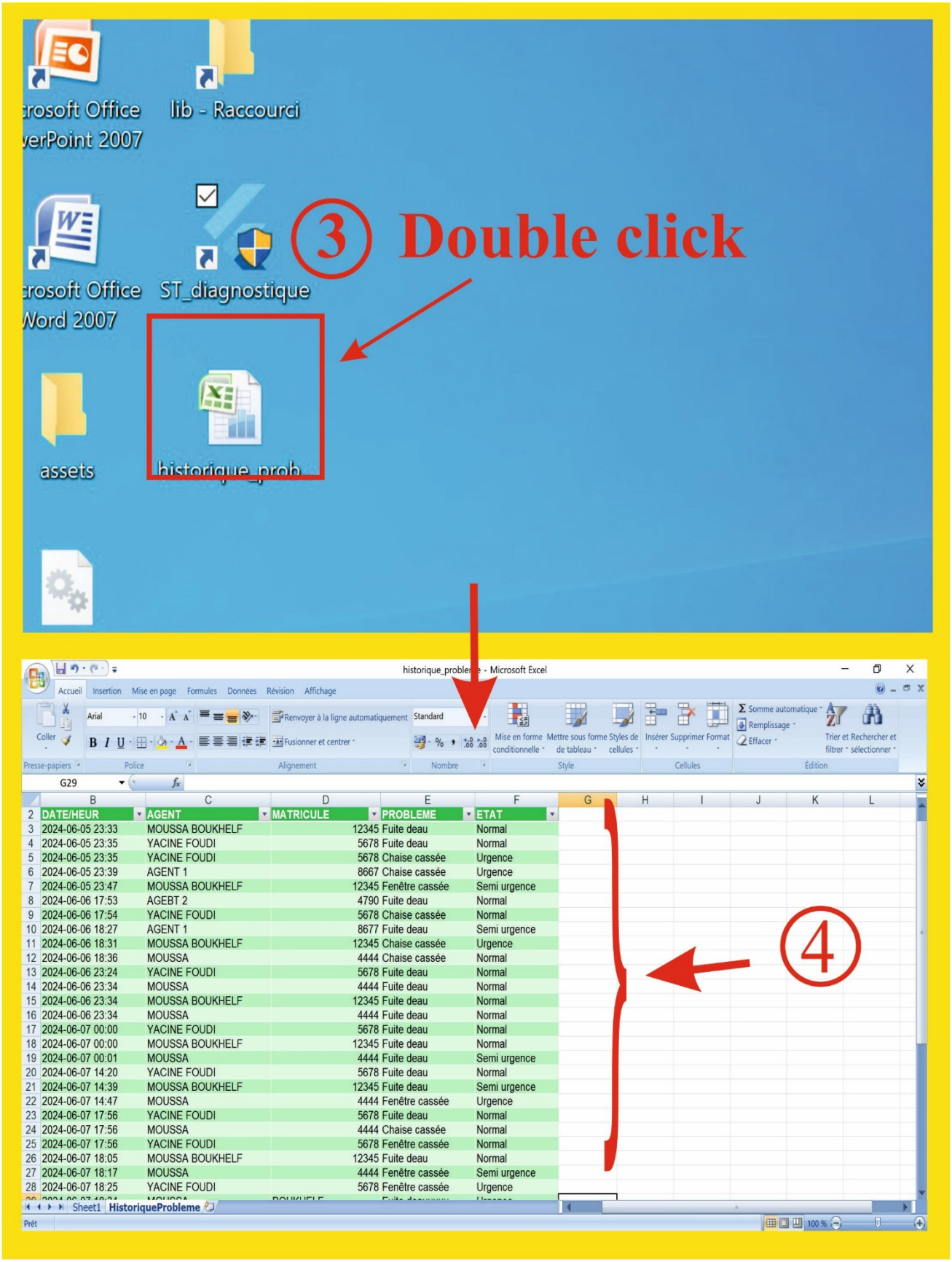


Figure III. 21 : Exporté l'historique vers Excel.

Une fois l'historique des problèmes signalés exporté vers un fichier Excel, on peut le déplacer sur le bureau pour plus de commodité. Pour l'ouvrir dans Excel, il vous suffit de double-cliquer sur le fichier Excel sur votre bureau, ce qui ouvrira Excel et chargera le fichier pour que on peut le consulter et le modifier si nécessaire.



DATE/HEUR	AGENT	MATRICULE	PROBLEME	ETAT
2024-06-05 23:33	MOUSSA BOUKHELF	12345	Fuite deau	Normal
2024-06-05 23:35	YACINE FOUDI	5678	Fuite deau	Normal
2024-06-05 23:35	YACINE FOUDI	5678	Chaise cassée	Urgence
2024-06-05 23:39	AGENT 1	8667	Chaise cassée	Urgence
2024-06-05 23:47	MOUSSA BOUKHELF	12345	Fenêtre cassée	Semi urgence
2024-06-06 17:53	AGEBT 2	4790	Fuite deau	Normal
2024-06-06 17:54	YACINE FOUDI	5678	Chaise cassée	Normal
2024-06-06 18:27	AGENT 1	8677	Fuite deau	Semi urgence
2024-06-06 18:31	MOUSSA BOUKHELF	12345	Chaise cassée	Urgence
2024-06-06 18:36	MOUSSA	4444	Chaise cassée	Normal
2024-06-06 23:24	YACINE FOUDI	5678	Fuite deau	Normal
2024-06-06 23:34	MOUSSA	4444	Fuite deau	Normal
2024-06-06 23:34	MOUSSA BOUKHELF	12345	Fuite deau	Normal
2024-06-06 23:34	MOUSSA	4444	Fuite deau	Normal
2024-06-07 00:00	YACINE FOUDI	5678	Fuite deau	Normal
2024-06-07 00:00	MOUSSA BOUKHELF	12345	Fuite deau	Normal
2024-06-07 00:01	MOUSSA	4444	Fuite deau	Semi urgence
2024-06-07 14:20	YACINE FOUDI	5678	Fuite deau	Normal
2024-06-07 14:39	MOUSSA BOUKHELF	12345	Fuite deau	Semi urgence
2024-06-07 14:47	MOUSSA	4444	Fenêtre cassée	Urgence
2024-06-07 17:56	YACINE FOUDI	5678	Fuite deau	Normal
2024-06-07 17:56	MOUSSA	4444	Chaise cassée	Normal
2024-06-07 17:56	YACINE FOUDI	5678	Fenêtre cassée	Normal
2024-06-07 18:05	MOUSSA BOUKHELF	12345	Fuite deau	Normal
2024-06-07 18:17	MOUSSA	4444	Fenêtre cassée	Semi urgence
2024-06-07 18:25	YACINE FOUDI	5678	Fenêtre cassée	Urgence

Figure III. 22 : Historique des problémes signalé sous forme d'un fichier Excel . .

III.7 Perspective :

Dans cette partie, nous allons fournir des suggestions future pour améliorer notre application à destination des étudiantes intéressées par le développement d'applications.

A. La maquette 3D :

Définition : Selon le dictionnaire « Larousse » une maquette est : "Une représentation en deux ou trois dimensions, le plus souvent à échelle réduite, ou agrandie mais fidèle dans ses proportions, d'une construction, d'un appareil, d'un décor, d'un objet quelconque. Elle peut aussi être un ensemble de pièces à monter pour obtenir une reproduction en modèle réduit de quelque chose, en particulier d'un avion, d'un véhicule : monter une maquette" **[12].**

Modélisation de la maquette :

Pour la conception de cette maquette, nous avons opté pour AutoCAD, un logiciel de conception assisté par ordinateur qui nous a facilité la création de notre maquette en 3D.

Ci-dessous, nous présentons les étapes de modélisation de la maquette de notre faculté.

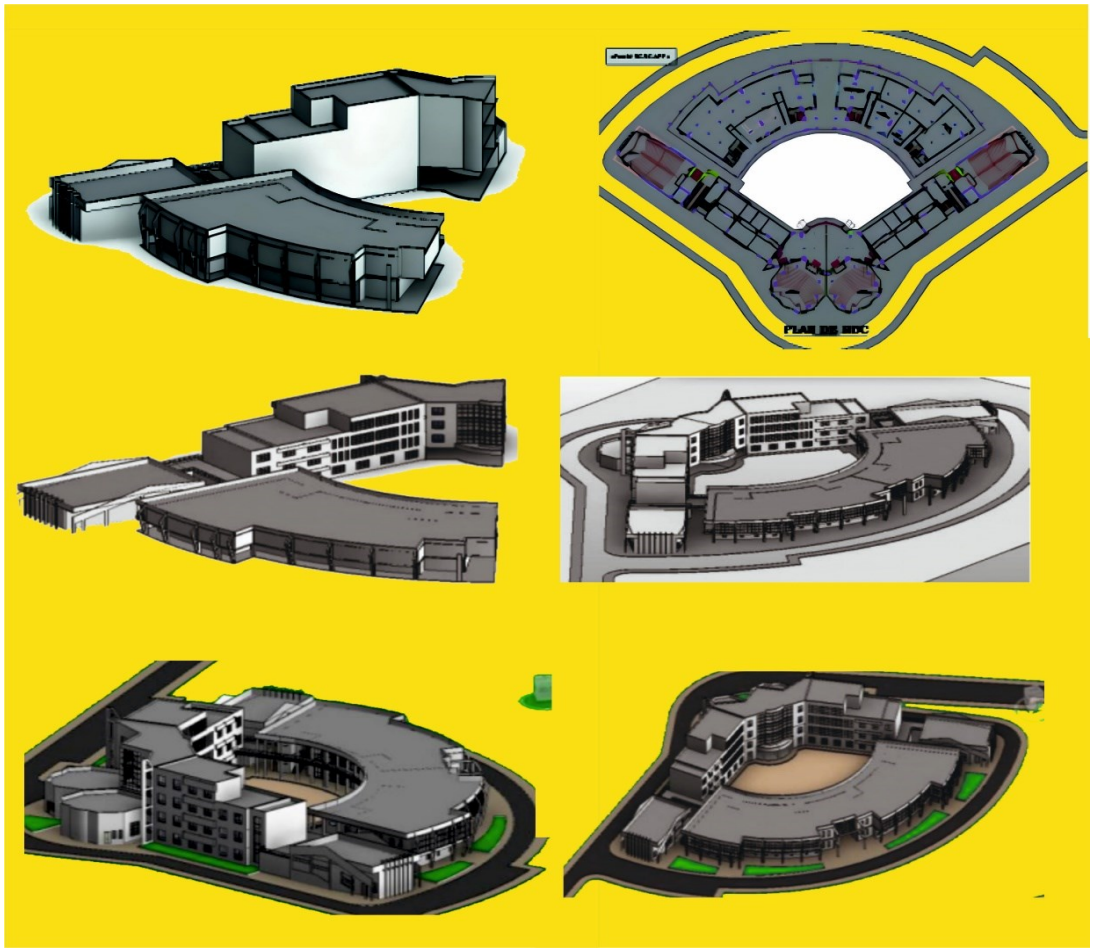


Figure III. 23 : La maquette 3D de notre faculté.

Nous proposons des idées d'amélioration et d'intégration de maquettes dans votre application, voici quelques suggestions :

- **Intégration des Maquettes 3D :**

  Visualisation des Problèmes : Offre la possibilité aux utilisateurs de visualiser les problèmes signalés directement sur une maquette 3D de la faculté. Les agents pourraient cliquer sur différentes sections de la maquette pour signaler ou vérifier des problèmes.

  Navigation Interactive : Incorpore une fonctionnalité de navigation interactive dans les maquettes 3D pour permettre aux utilisateurs de mieux comprendre les emplacements des problèmes signalés et de se déplacer virtuellement dans l'environnement de la faculté.

- **Amélioration de l'Interface Utilisateur :**

  Design Intuitif : Utilise des maquettes 3D pour améliorer l'interface utilisateur en rendant la navigation et la signalisation de problèmes plus intuitives. Par exemple, des représentations visuelles des différents départements et lieux pourraient simplifier l'identification des zones à problèmes.

  Animations et Transitions : Intégrez des animations et des transitions fluides basées sur les maquettes 3D pour rendre l'expérience utilisateur plus engageante.

- **Analyse et Reportions :**

  Visualisation des Données : Utilise des maquettes 3D pour visualiser les données de l'historique des problèmes signalés, aidant à identifier les tendances et les zones à haute fréquence de problèmes.

  Rapports Dynamiques : Crée des rapports dynamiques qui utilisent des maquettes 3D pour illustrer les statistiques et les analyses de manière plus visuelle et compréhensible.

Remarque : L'ajout de maquettes 3D dans notre application peut améliorer l'expérience utilisateur et fournir des outils efficaces pour visualiser, communiquer et analyser les problèmes signalés. Cela rendra notre application plus attrayante et fonctionnelle.

Capture d'écran :

Appuyé sur la notification qui affiche dans la maquette suivante :

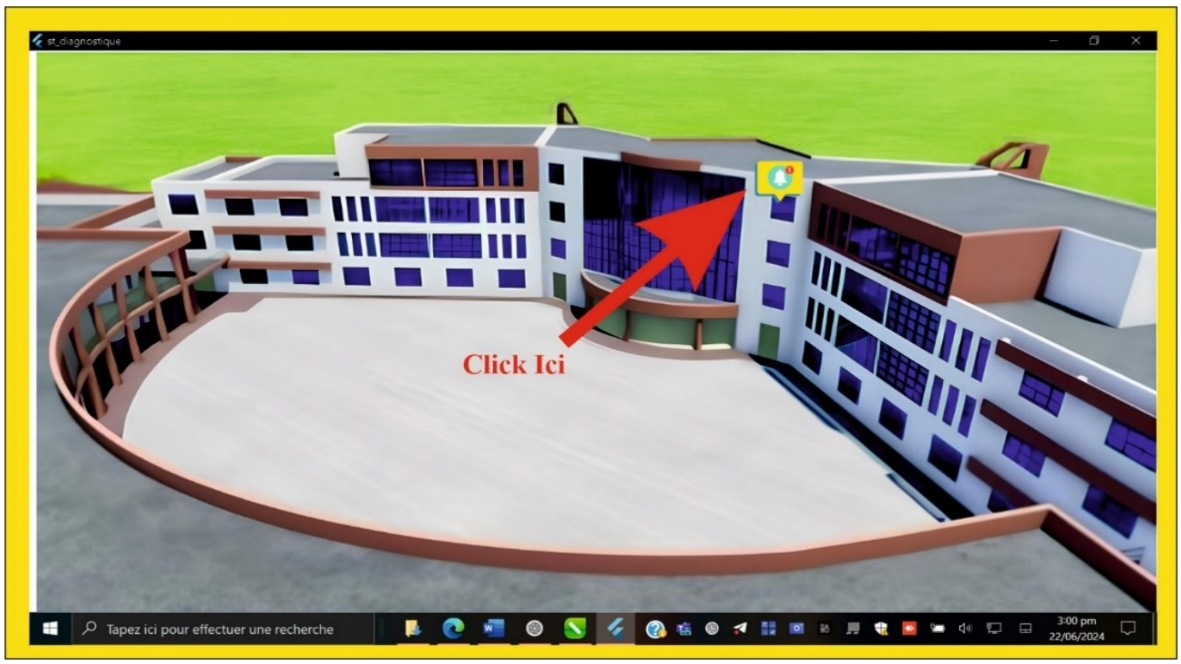


Figure III. 24 : Notification dans la maquette.

Afficher le problème signalé par l'agent avec ses informations détaillées.

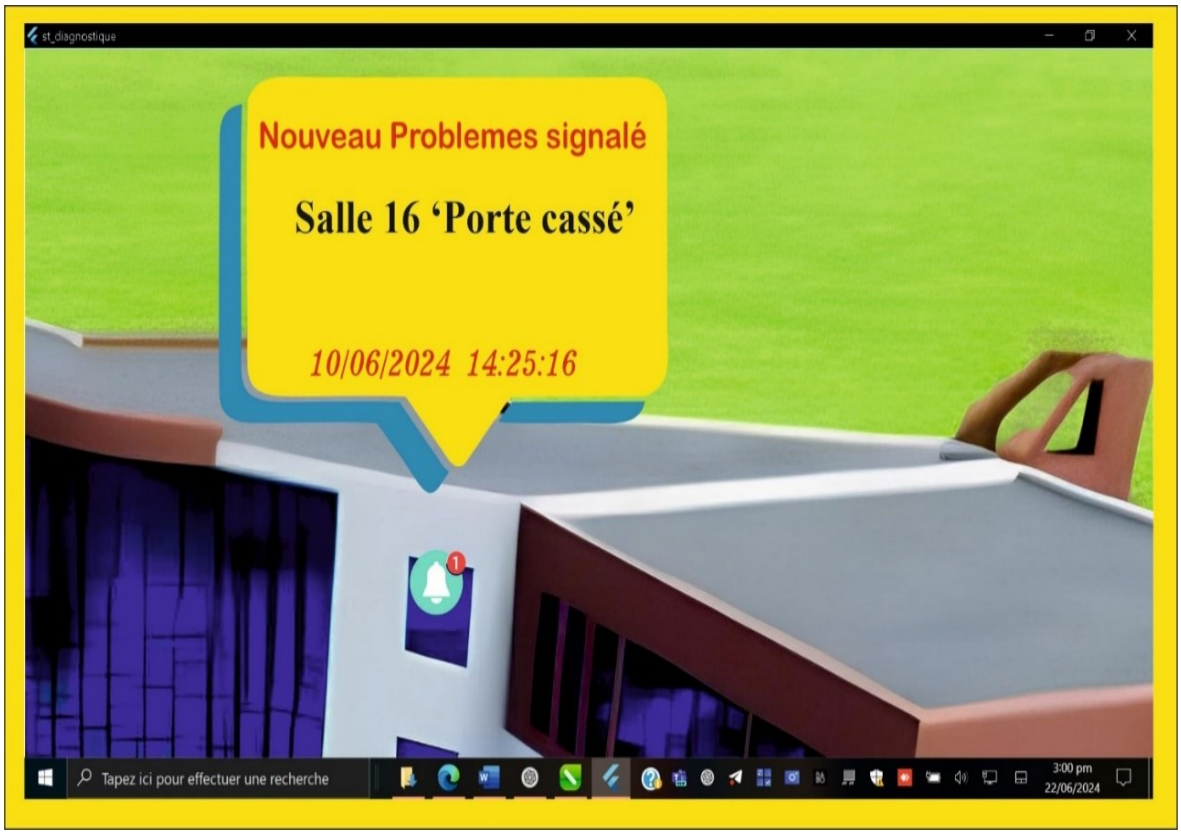


Figure III. 25 : Affichage de Problème .

B. Chat : Ajouter nouvelle Option de communication entre agent et superviseur

Explication : La nouvelle fonction de communication directe entre l'agent et le superviseur :

Amélioration de l'interaction et de l'efficacité. Dans le but de simplifier l'utilisation de l'application et de renforcer la communication entre le superviseur et l'agent, nous avons ajouté une fonctionnalité de communication directe entre les deux parties. Cette fonctionnalité permet au superviseur d'attribuer des tâches à l'agent en envoyant des messages directement dans l'application. Par exemple, le superviseur peut envoyer des instructions détaillées, des délais et des priorités pour chaque tâche, ce qui facilite la compréhension et l'exécution des tâches par l'agent. De plus,

Cette fonctionnalité permet à l'agent de poser des questions ou de demander des clarifications sur les tâches assignées, assurant ainsi une meilleure compréhension des attentes et une exécution plus efficace des tâches. En intégrant cette option de communication directe, nous visons à améliorer la coordination, la collaboration et l'efficacité globale du processus de gestion des problèmes signalés.

III.8 Conclusion :

Les résultats obtenus du développement de notre application montrent dans l'ensemble qu'il sont globalement très satisfaisant . l'usage et l'explication du fichier Excel eu qualité de base de données va permettre à terme d'améliorer le suivi et le diagnostic des fonctionnement au niveau de notre faculté

Conclusion Générale

Ce projet nous a permis de concevoir, développer et intégrer une application mobile sous Android, une plateforme dominante sur le marché des terminaux mobiles. La réalisation de ce projet s'est déroulée en trois étapes principales :

- Nous avons commencé par une analyse des différents systèmes d'exploitation, en nous concentrant sur Android, ses versions et ses outils de développement.
- Puis en à passé à concevoir notre application, nous avons utilisé le langage de programmation Flutter et logiciel Android Studio comme environnement de développement.
- Enfin , nous avons réussi à créer un outil simple et efficace permettant aux utilisateurs de suivre de manière numérique les problèmes de la Faculté des Sciences et Sciences Appliquées.

Bien que l'outil réalisé soit très satisfaisant, plusieurs améliorations restent possibles, telles que l'intégration de maquettes 3D et l'amélioration de la communication entre le superviseur et l'agent. Cependant, certains obstacles ont entravé l'amélioration complète de notre application :-

- La distance maximale de connexion au réseau est limitée (30 metre)
- Nombre d'utilisateurs ,L'application ne peut actuellement être utilisée que par huit personnes (un superviseur et sept agents).

Ces contraintes ont limité notre capacité à améliorer l'application de manière exhaustive. Nous laissons ces défis aux étudiants futurs qui choisiront de poursuivre cette étude, en espérant qu'ils trouveront des solutions innovantes pour surmonter ces obstacles et améliorer notre travail.

Références bibliographiques

[1] www.Application mobile : définition - Lexique des entreprises (infonet.fr).com **consulté le 10 Juin 2024**

[2] www.codeur.com/blog/application-mobile-definition-conseils-et-exemples/ .com **consulté le 10 Juin 2024**

[3] www. Système d'exploitation d'un smartphone : rôle, classification et critères de choix | Blog Les Jeudis.com **consulté le Juin 2024**

[4] www.pmtic.net/contenu-en-ligne/environnement-numerique/mobile/smartphones/recapitulons/systemes-d-exploitation-0 **com consulté le 10 Juin 2024**

[5] www.Android génère des opportunités.com **consulté le 10 Juin 2024**

[6] www.243tech.com/naissance-et-evolution-android/ .com **consulté le 10 Juin 2024**

[7] www.aws.amazon.com/fr/what-is/sdk/.com **consulté le 10 Juin 2024**

[8] www**.**Android Studio — Wikipédia (wikipedia.org) .com **consulté le 10 Juin 2024**

[9] www.Flutter (logiciel) — Wikipédia (wikipedia.org) .com **consulté le 10 Juin 2024**

[10] www.Make Windows desktop apps | Flutter .com **consulté le 10 Juin 2024**

[11] www.Dart overview | Dart .com **consulté le 10 Juin 2024**

[12] www.larousse.fr. **consulté le 10 Juin 2024**

[13] DERRADJI Lyes et BENTOUTAH Sarah , « Etude et réalisation d'une maquette 3D de la FSSA pour le diagnostic et le suivi » , mémoire de master, faculté des sciences et des sciences appliquées université de Bouira , 2022 .

Annexes :

ANEXE 1 :l'acces pour admin

```
import 'package:flutter/material.dart';
import 'package:st_mainten/adminpage.dart';


class AccesAdmin extends StatelessWidget {
  // const AccesAdmin({super.key});
  TextEditingController _usernameController = TextEditingController();
  TextEditingController _passwordController = TextEditingController();
  @override
  Widget build(BuildContext context) {
    return Scaffold(
      body: Padding(
        padding: EdgeInsets.all(16.0),
        child: SingleChildScrollView(
          child: Column(
            mainAxisAlignment: MainAxisAlignment.center,
            children: [
              SizedBox(
                height: 50,
              ),
              Text(
                'Veuillez saiair votre nom et votre mot de passe,s il veut
plait',
                style: TextStyle(
                    fontSize: 25,
                    color: Colors.black,
                    fontStyle: FontStyle.italic),
              ),
```

```
            SizedBox(height: 50),
            TextField(
              controller: _usernameController,
              style: TextStyle(fontSize: 18),
              decoration: InputDecoration(
                  labelText: 'Nom et prenom',
                  labelStyle: TextStyle(fontSize: 20),
                  hintText: 'Entrez votre nom et prenom',
                  icon: Icon(
                    Icons.person,
                    color: Colors.blue,
                    size: 25,
                  )),
              keyboardType: TextInputType.text,
            ),
            TextField(
              controller: _passwordController,
              style: TextStyle(fontSize: 18),
              decoration: InputDecoration(
                  labelText: 'Password',
                  hintText: 'Entrez votre password',
                  labelStyle: TextStyle(fontSize: 20),
                  icon: Icon(
                    Icons.lock,
                    color: Colors.red,
                    size: 25,
                  )),
            ),
            SizedBox(height: 200),
            ElevatedButton(
```

```
                onPressed: () {

                    // mainAxisAlignment: MainAxisAlignment.spaceBetween,
                    children: [
                      Icon(
                        Icons.list,
                        color: Colors.black,
                        size: 30,
                      ),
                      SizedBox(width: 16), // Icône à gauche du texte du
bouton
                      Text(
                        'Afficher types',
                        style: TextStyle(fontSize: 18, color:
Colors.white),
                      ),
                    ],
                  ),
                  style: ButtonStyle(
                      minimumSize: MaterialStateProperty.all(Size(300,
50)),
                      backgroundColor:
                          MaterialStateProperty.all<Color>(Colors.blue)),
                ),
                SizedBox(height: 40),
                ElevatedButton(
                  onPressed: () {
                    var route = MaterialPageRoute(
                        builder: (BuildContext context) => Listlieu());
                    Navigator.of(context).push(route);
                  },
                  child: Row(
                    // mainAxisAlignment: MainAxisAlignment.spaceBetween,
                    children: [
                      Icon(
                        Icons.list,
                        color: Colors.black,
                        size: 30,
                      ),
                      SizedBox(width: 16), // Icône à gauche du texte du
bouton
                      Text(
                        'Afficher lieux',
                        style: TextStyle(fontSize: 18, color:
Colors.white),
                      ),
                    ],
                  ),
                  style: ButtonStyle(
                      minimumSize: MaterialStateProperty.all(Size(300,
50)),
                      backgroundColor:
                          MaterialStateProperty.all<Color>(Colors.blue)),
                ),
                SizedBox(height: 40),
              ],
            ),
          ),
        )));
  }
}
```

ANEXE 4 : page d'agent

```
import 'package:flutter/material.dart';
import 'package:st_mainten/ajouterprobleme.dart';
import 'package:st_mainten/listproblemes.dart';
import 'package:st_mainten/accessajent.dart';
import 'package:st_mainten/sqldb.dart';
import 'dart:typed_data';

class Agentpage extends StatelessWidget {
  // const Agentpage({Key? key});
  String? username = AccesAjent.username;
  String? password = AccesAjent.password;
  Databasehelper ob = Databasehelper();
  Future<List<Map<String, dynamic>>> getAllinfoagents() async {
    List<Map<String, dynamic>> result = await ob.getData(
        "SELECT * FROM infoagents WHERE Nom = '$username' AND Motdepasse =
'$password'");
    return result;
  }

  @override
  Widget build(BuildContext context) {
    return Scaffold(
      appBar: AppBar(
        backgroundColor: Colors.blue,
        title: Text(
          'Agent page',
          style: TextStyle(fontSize: 30, color: Colors.white),
        ),
      ),
      body: Container(
        margin: EdgeInsets.all(55),
        child: Center(
          child: FutureBuilder(
            future: getAllinfoagents(),
            builder: (BuildContext context,
                AsyncSnapshot<List<Map<String, dynamic>>> snapshot) {
              if (snapshot.connectionState == ConnectionState.waiting) {
                return CircularProgressIndicator();
              } else if (snapshot.hasError) {
                return Text('Error: ${snapshot.error}');
              } else {
                // Récupérer l'image à partir des données de la base de
données
                Uint8List? imageBytes;
                if (snapshot.data != null && snapshot.data!.isNotEmpty) {
                  // Assurez-vous que 'image' est le nom de la colonne
contenant les données d'image
                  imageBytes = snapshot.data![0]['image'];
                }

                return Column(
                  children: [
                    CircleAvatar(
                      radius: 80,
                      // Utilisez MemoryImage pour afficher les données
d'image
```

```
                      backgroundImage:
                          imageBytes != null ? MemoryImage(imageBytes) :
null,
                    ),
                    SizedBox(height: 40),
                    ElevatedButton(
                      onPressed: () {
                        var route = MaterialPageRoute(
                            builder: (BuildContext context) =>
                                Ajouterprobleme());
                        Navigator.of(context).push(route);
                      },
                      child: Row(
                        children: [
                          Icon(
                            Icons.add,
                            color: Colors.white,
                            size: 30,
                          ),
                          Icon(
                            Icons.report_problem,
                            color: Colors.red,
                            size: 30,
                          ),
                          SizedBox(width: 16),
                          Expanded(
                            child: Text(
                              'Déclarer un problème',
                              style:
                                  TextStyle(fontSize: 20, color:
Colors.white),
                            ),
                          ),
                        ],
                      ),
                      style: ElevatedButton.styleFrom(
                        shape: RoundedRectangleBorder(
                            borderRadius: BorderRadius.zero),
                        minimumSize: Size(300, 50),
                        backgroundColor: Colors.blue,
                      ),
                    ),
                    SizedBox(height: 30),
                    ElevatedButton(
                      onPressed: () {
                        Navigator.of(context).pushAndRemoveUntil(
                            MaterialPageRoute(
                                builder: (context) => Listproblemes()),
                            (route) => true);
                      },
                      child: Row(
                        children: [
                          Icon(
                            Icons.screenshot_monitor,
                            color: Colors.white,
                            size: 30,
                          ),
                          SizedBox(width: 16),
                          Expanded(
                            child: Text(
                              'Afficher problèmes',
```

```
                                style:
                                    TextStyle(fontSize: 20, color:
Colors.white),
                                ),
                            ),
                          ],
                        ),
                        style: ElevatedButton.styleFrom(
                          shape: RoundedRectangleBorder(
                              borderRadius: BorderRadius.zero),
                          minimumSize: Size(300, 50),
                          backgroundColor: Colors.blue,
                        ),
                      ),
                    ],
                  );
                }
              },
            ),
          ),
        ),
      ),
    );
  }
}
```

ANEXE 5 :ajouter agent

```
import 'package:flutter/material.dart';
import 'package:st_mainten/listagents.dart';
import 'package:st_mainten/sqldb.dart';
import 'package:image_picker/image_picker.dart';
import 'dart:io';
import 'dart:typed_data';
import 'dart:convert';
import 'package:image/image.dart' as img;
import 'package:web_socket_channel/web_socket_channel.dart';
import 'package:network_info_plus/network_info_plus.dart';

class Ajoutagent1 extends StatefulWidget {
  const Ajoutagent1({super.key});
  @override
  State<Ajoutagent1> createState() => _Ajoutagent1State();
}

class _Ajoutagent1State extends State<Ajoutagent1> {
  Databasehelper ob = Databasehelper();
  String imagepath = "";
  ImagePicker picker = ImagePicker();
  TextEditingController _Nom = TextEditingController();
  TextEditingController _Motdepasse = TextEditingController();
  void getimage() async {
    var pickedimage = await picker.pickImage(source: ImageSource.gallery);
    if (pickedimage != null) {
      setState(() {
        imagepath = pickedimage.path;
      });
    }
  }

  Future<void> sendDataViaWebSocket() async {
```

```
    try {
      //  String? localIp = await getLocalIpAddress();
      // String? serverUrl =
'ws://<adresse_ip_du_serveur>:<port_du_serveur>'; // Remplacez par
l'adresse IP et le port de votre serveur WebSocket
      String serverUrl = 'ws://192.168.8.250:8080';
      //String serverUrl = 'ws://$localIp:8080';
      if (serverUrl == null) {
        print('Impossible de récupérer l\'URL du serveur WebSocket');
        return;
      }

      File imageFile = File(imagepath); // Chemin vers votre fichier image
      Uint8List compressedImageBytes = await compressImage(imageFile);
      List<int> imageIntList = compressedImageBytes.toList();
      Map<String, dynamic> data = {
        'nom': _Nom.text,
        'motDePasse': _Motdepasse.text,
        'imageBytes': imageIntList, // Utiliser la liste d'entiers pour
l'image
      };

      String jsonData = jsonEncode(data); // Convertir les données en JSON

      WebSocketChannel channel =
WebSocketChannel.connect(Uri.parse(serverUrl));
      channel.sink.add(jsonData);

      print('Données envoyées via WebSocket avec succès');
      channel.sink.close(); // Fermer la connexion après l'envoi
    } catch (e) {
      print('Erreur lors de l\'envoi des données via WebSocket : $e');
    }
  }

  String _ipAddress = 'Unknown';

  Future<void> _getLocalIPAddress() async {
    final info = NetworkInfo();
    String? ipAddress;

    try {
      ipAddress =
          await info.getWifiIP(); // Get IP address of the Wi-Fi interface
    } catch (e) {
      print('Failed to get IP address: $e');
      ipAddress = 'Failed to get IP address';
    }

    setState(() {
      _ipAddress = ipAddress ?? 'Not connected';
      print(_ipAddress);
    });
  }

  Future<Uint8List> compressImage(File imageFile) async {
    // Lire les bytes de l'image depuis le fichier
    Uint8List imageBytes = await imageFile.readAsBytes();
    // Décodez les bytes de l'image en une image utilisable avec la
bibliothèque image
    img.Image? image = img.decodeImage(imageBytes);
```

```
    if (image != null) {
                                ),
                              );
                            }
                            return SizedBox.shrink();
                          },
                        ),
                      ),
                      Visibility(
                        visible: departementschoise != null,
                        child: Container(
                          margin: EdgeInsets.fromLTRB(0, 10, 0, 10),
                          color: Colors.blue,
                          padding: EdgeInsets.only(left: 8),
                          child: FutureBuilder(
                            future: getAlllieutypes(),
                            builder: (ctx,
                                AsyncSnapshot<List<Map<String, dynamic>>>
snp) {
                              if (snp.hasData) {
                                List<Map<String, dynamic>> listtypes =
snp.data!;
                                return DropdownButtonHideUnderline(
                                  child: DropdownButton(
                                    isExpanded: true,
                                    icon: Icon(
                                      Icons.arrow_drop_down,
                                      color: Colors.white,
                                      size: 30,
                                    ),
                                    hint: Row(
```

```
                                    children: [

import 'package:st_mainten/listlieu.dart';


class Updatelieu extends StatefulWidget {
  final id;
  final type;
  final nombre;
  const Updatelieu({Key? key, this.id, this.type, this.nombre})
      : super(key: key);


  @override
  State<Updatelieu> createState() => _UpdatelieuState();
}


class _UpdatelieuState extends State<Updatelieu> {
  Databasehelper ob = Databasehelper();
  TextEditingController _Type = TextEditingController();
  TextEditingController _Nombre = TextEditingController();
  @override
  void initState() {
    _Type.text = widget.type;
    _Nombre.text = widget.nombre;
    // TODO: implement initState
    super.initState();
  }


  @override
  Widget build(BuildContext context) {
    return Scaffold(
      appBar: AppBar(
```

```
        backgroundColor: Colors.blue,
        title: Text(
          'Update lieu',
          style: TextStyle(color: Colors.white, fontSize: 30),
        ),
      ),
      body: Container(
        margin: EdgeInsets.all(10),
        child: Column(children: [
          SizedBox(height: 30),
          TextField(
            controller: _Type,
            style: TextStyle(fontSize: 18),
            decoration: InputDecoration(
              border: OutlineInputBorder(
                borderRadius: BorderRadius.only(
                    topLeft: Radius.circular(20.0),
                    topRight: Radius.circular(0.0),
                    bottomLeft: Radius.circular(20.0),
                    bottomRight: Radius.circular(0.0)),
              ),
              labelText: 'Type',
              labelStyle: TextStyle(fontSize: 20),
              hintText: 'Entrer le Type',
              prefixIcon: Icon(Icons.add, color: Colors.blue, size: 25),
            ),
            keyboardType: TextInputType.text,
          ),
          SizedBox(height: 30),
          TextField(
```

```
            controller: _Nombre,
            style: TextStyle(fontSize: 18),
            decoration: InputDecoration(
              border: OutlineInputBorder(
                borderRadius: BorderRadius.only(
                    topLeft: Radius.circular(20.0),
                    topRight: Radius.circular(0.0),
                    bottomLeft: Radius.circular(20.0),
                    bottomRight: Radius.circular(0.0)),
              ),
              labelText: 'Numéro',
              labelStyle: TextStyle(fontSize: 20),
              hintText: 'Entrer le numéro',
              prefixIcon: Icon(Icons.add, color: Colors.blue, size: 25),
            ),
            keyboardType: TextInputType.text,
          ),
          SizedBox(height: 40),
          ElevatedButton(
            onPressed: () async {
              int rep = await ob.updateData('''
              UPDATE "lieu"SET
              Type="${_Type.text}",
              Nombre="${_Nombre.text}"
              WHERE id="${widget.id}"
              ''');
              if (rep > 0) {
                Navigator.of(context).pushAndRemoveUntil(
                    MaterialPageRoute(builder: (context) => Listlieu()),
                    (route) => true);
```

```
              }
            },
            child: Text(
              "modifier",
              style: TextStyle(fontSize: 20, color: Colors.white),
            ),
            style: ElevatedButton.styleFrom(
              shape: RoundedRectangleBorder(borderRadius:
BorderRadius.zero),
              minimumSize: Size(150, 50),
              backgroundColor: Colors.blue,
            ),
          ),
        ]),
      ),
    );
  }
}
```

ANEXE 20 :mise à jours des type de lieu

```
import 'package:flutter/material.dart';
import 'package:st_mainten/listlieutypes.dart';
import 'package:st_mainten/sqldb.dart';

class Updatelieutype extends StatefulWidget {
  final id;
  final type;
  const Updatelieutype({Key? key, this.id, this.type}) : super(key: key);

  @override
  State<Updatelieutype> createState() => _UpdatelieutypeState();
```

```
}


class _UpdatelieutypeState extends State<Updatelieutype> {
  Databasehelper ob = Databasehelper();
  TextEditingController _Type = TextEditingController();
  @override
  void initState() {
    _Type.text = widget.type;
    // TODO: implement initState
    super.initState();
  }


  @override
  Widget build(BuildContext context) {
    return Scaffold(
      appBar: AppBar(
          title: Text(
            "Update type",
            style: TextStyle(color: Colors.white, fontSize: 30),
          ),
          backgroundColor: Colors.blue),
      body: Container(
        margin: EdgeInsets.all(10),
        child: Column(children: [
          SizedBox(height: 30),
          TextField(
            controller: _Type,
            style: TextStyle(fontSize: 18),
            decoration: InputDecoration(
              border: OutlineInputBorder(
```

```
                borderRadius: BorderRadius.only(
                    topLeft: Radius.circular(20.0),
                    topRight: Radius.circular(0.0),
                    bottomLeft: Radius.circular(20.0),
                    bottomRight: Radius.circular(0.0)),
              ),
              labelText: 'Type',
              labelStyle: TextStyle(fontSize: 20),
              hintText: 'Entrer le type ',
              prefixIcon: Icon(Icons.add, color: Colors.blue, size: 25),
            ),
            keyboardType: TextInputType.text,
          ),
          SizedBox(height: 40),
          ElevatedButton(
            onPressed: () async {
              int rep = await ob.updateData('''
              UPDATE "lieutypes"SET
              Type="${_Type.text}"
              WHERE id="${widget.id}"
              ''');
              if (rep > 0) {
                Navigator.of(context).pushAndRemoveUntil(
                    MaterialPageRoute(builder: (context) =>
Listlieutypes()),
                    (route) => true);
              }
            },
            child: Text(
              "modifier",
              style: TextStyle(fontSize: 20, color: Colors.white),
```

ملخص

الهدف من مشروع نهاية الدراسة هذا هو تزويد قسم الصيانة التابع لـ FSSA بجامعة البويرة بتطبيق حاسوبي لدعم هذه الخدمة في الإدارة والمراقبة اليومية وتشخيص القيود التي تمت مواجهتها على أسس FSSA: تسرب المياه، النوافذ المكسورة والمكاتب المتضررة ;تم تضمين هذا التطبيق على الهواتف الذكية التي تعمل بنظام التشغيل Android ويمكن أيضًا تثبيته على أجهزة الكمبيوتر الشخصية (الكمبيوتر المحمول أو سطح المكتب). سيسمح استغلال هذا التطبيق في النهاية بالإدارة الفعالة والمراقبة الدائمة للمشاكل التي يتم الإبلاغ عنها في الوقت الفعلي، والتي سيكون تأثيرها واضحًا من حيث معالجة المشكلات المثارة، وستسمح وسائل تكنولوجيا المعلومات الحديثة هذه أيضًا بتحسين إدارة بعض المشكلات التي أبلغت عنها الكلية، وبالتالي تساعد على تحسين جودة الخدمات المقدمة.

Résumé

L'objectif de ce PFE est la mise à disposition du service maintenance de la FSSA de l'université de Bouira d'une application informatique , pour le soutenu de ce service à la gestion, le suivi quotidien et le diagnostic des contrainte rencontrées sur le terrain de la FSSA : fuite d'eau , fenêtre cassées , bureaux endommagé,Cette application est embarquée sur , des téléphones smartphones dont le système d'exploitation android ,Elle peut aussi être installée sur des pc (portable ou bureau) .L'exploitation de cette application va permettre à terme gestion efficace , un suivis permanent des problèmes signalés eu temps réel , dont l'impact sera visible au niveau de la prise en charge des problèmes soulevés , ce moyen informatique moderne va permettre également une amélioration dans la gestion d'une partie des problèmes signalés de la faculté , et d'aider donc à une meilleure qualité du services rendus .

Abstract

The objective of this End of Study Project is to provide the maintenance department of the FSSA of the University of Bouira with a computer application, to support this service in the management, daily monitoring and diagnosis of constraints encountered on the FSSA grounds: water leak, broken windows, damaged offices ,This application is embedded on smartphones with the Android operating system. It can also be installed on PCs (laptop or desktop). The exploitation of this application will ultimately allow effective management, permanent monitoring of problems reported in real time, the impact of which will be visible in terms of the handling of the problems raised, this modern IT means will also allow an improvement in the management of some of the problems reported by the faculty, and therefore help to improve the quality of the services provided.