

# RAPPORT DE STAGE D'ÉTÉ

---

## Test d'une Application Mobile ProLab

---

Réalisé au sein de l'entreprise Sama Consulting



*Par*  
MOHAMED AZIZ KAROUI

Spécialité : Licence Fondamentale en Sciences de l'Informatique

Année Universitaire : 2019-2020

# **Remerciements**

Je tiens à remercier toutes les personnes qui ont contribué au succès de mon stage et qui m'ont aidé lors de la rédaction de ce rapport.

Tout d'abord, j'adresse mes remerciements à monsieur, Mohamed Ali Adala qui m'a beaucoup aidé dans ma recherche de stage et m'a permis de postuler dans cette entreprise. Son écoute et ses conseils m'ont permis de cibler mes candidatures, et de trouver ce stage qui était en totale adéquation avec mes attentes.

Je tiens à remercier vivement mon maître de stage, Sofien Beji, Consultant de système au sein de l'entreprise Sama Consulting , pour son accueil, le temps passé ensemble et le partage de son expertise au quotidien. Grâce aussi à sa confiance j'ai pu m'accomplir totalement dans mes missions. Il fut d'une aide précieuse dans les moments les plus délicats.

Je remercie également toute l'équipe de travail pour leur accueil, leur esprit d'équipe.

Enfin, je tiens à remercier toutes les personnes qui m'ont conseillé et relu lors de la rédaction de ce rapport de stage.

# Table des matières

<b>Remerciements</b>	<b>i</b>
<b>Introduction Générale</b>	<b>1</b>
<b>1 Cadre général du Stage</b>	<b>2</b>
1.1 Introduction . . . . .	2
1.2 Présentation générale de l'organisme d'accueil . . . . .	2
1.2.1 Présentation de la société Sama Consulting . . . . .	2
1.2.2 Les membres de Sama Consulting . . . . .	2
1.3 Les Buts de stage . . . . .	3
1.3.1 Compétences à apprendre . . . . .	3
1.3.2 Appliquer ces compétences dans un projet spécifié . . . . .	4
1.4 Conclusion . . . . .	4
<b>2 Préparation des Tests</b>	<b>5</b>
2.1 Introduction . . . . .	5
2.2 Les Test Cases et le test manuelle . . . . .	5
2.2.1 Préparation des Test Cases . . . . .	5
2.2.2 Appliquer sur un Environnement manuel . . . . .	7
2.3 Conclusion . . . . .	12
<b>3 L'automation du Test</b>	<b>13</b>
3.1 Introduction . . . . .	13
3.2 Implémenter Android Studio et Espresso . . . . .	13
3.2.1 Définition d'Espresso . . . . .	13
3.2.2 Installation de l'environnement Android Studio et son Framework Espresso . . . . .	16
3.3 Créer une Application de Test . . . . .	17
3.4 Faire des Test automatique sur ProLab Mobile 5 . . . . .	23
3.5 Conclusion . . . . .	24
<b>Conclusion Générale</b>	<b>25</b>
<b>Bibliographie</b>	<b>26</b>

# Table des figures

1.1	Les membres de l'entreprise Sama Consulting . . . . .	3
1.2	Graphe de l'étude réalisé par Nuance Communications . . . . .	3
1.3	Logo d'ISTQB . . . . .	4
2.1	Le Guide Code Couleurs de l'application ProLab Mobile 5 . . . . .	5
2.2	Les Premiers Test Cases effectués . . . . .	6
2.3	Les Deuxièmes Test Cases effectués . . . . .	7
2.4	Les Troisièmes Test Cases effectués . . . . .	7
3.1	Logo du Framework Espresso . . . . .	13
3.2	Comparaison entre Espresso et Appium . . . . .	14
3.3	Espresso Cheatsheet . . . . .	15

# Introduction Générale

D'après l'étude réalisé par Nuance Communications sur « l'importance des applications mobiles pour les marques », 70% des sondés ont installé plus de 10 applications sur leur Smartphone et que les utilisateurs passent en moyenne 10% de temps supplémentaire sur leurs applications que sur le web.

« Sama Consulting » est une société d'ingénierie informatique qui comprend l'importance des applications mobile dans notre époque moderne et mise beaucoup sur cet aspect pour son développement.

Du 04 août 2020 au 04 septembre 2020, j'ai effectué un stage au sein de la société « Sama Consulting ». Obtenu grâce au directeur de l'entreprise « A.B Global Consulting » Mohamed Ali Adala où j'ai effectué mon 2018 stage d'observation là-dedans.

J'ai pris cette opportunité pour en apprendre plus sur la développement des projets et la vie pratique dans une société d'ingénierie informatique.

Mon stage a consisté à comprendre les étapes de développement d'un application mobile puis exécuter l'une de ces étapes en Intégration (Testing) sur un progiciel Prolab Mobile 5 déjà conçu par la société et ce présent rapport décrit les différentes étapes de mon travail et progression. Il s'articule autour trois chapitres :

- Le premier chapitre présente l'organisme d'accueil et les buts de stage.
- Le deuxième chapitre est consacré pour étudier l'application ProLab Mobile et apprendre à faire un "Case Testing" manuellement.
- Le troisième chapitre est consacré pour l'automation de ces "Case Testing" et apprendre à utilisé Android Studio et son Framework de test Espresso.
- Nous clôturons ce rapport par une conclusion générale qui récapitule le travail effectué et qui décrit certaines perspectives pour le projet réalisé.

# **Chapitre 1**

## **Cadre général du Stage**

### **1.1 Introduction**

Ce chapitre est consacré à présenter le cadre général du Stage. Il s'agit de présenter le contexte général du stage, de l'organisme d'accueil, ainsi que les tâches associés pendant le stage à apprendre pour réalisé le projet du stage.

### **1.2 Présentation générale de l'organisme d'accueil**

#### **1.2.1 Présentation de la société Sama Consulting**

Sama consulting [1] est une société d'ingénierie informatique spécialisée dans le développement de logiciels, la mise en place de systèmes d'information ainsi que le conseil et ce, dans différents domaines : santé, éducations, industrie, etc.

#### **1.2.2 Les membres de Sama Consulting**

Les Fondateurs de Sama consulting sont jeunes et avec des idées innovantes. Ils ont une bonne expérience dans la conduite de grands projets dans des domaines critiques comme les laboratoires ou les cliniques.

L'équipe de travail est composée à la fois de chefs de projets expérimentés, de jeunes ingénieurs et de techniciens, s'attache à concevoir et réaliser le meilleur applicatif au service de son utilisateur et de ses objectifs.

« Le professionnalisme, la réactivité et l'expertise » est le slogan de cette société.



FIGURE 1.1 – Les membres de l'entreprise Sama Consulting

## 1.3 Les Buts de stage

### 1.3.1 Compétences à apprendre

Avant tout, l'une des plus importantes buts de stage c'est apprendre à s'adapter à une nouvelle environnement sociale et être à l'aise là dans, de plus il y a les tâches manuelle pour réparer et prendre en soin du matériel de travail.

Ensuite, il y a les tâches pour apprendre les choses pratiques de ce domaine de développement et comme j'ai mentionné dans l'introduction général d'après l'étude de Nuance Communications [2] , il y a en moyenne 10% de gens qui utilise des SmartPhone qui passe leur temps supplémentaire sur leurs applications que sur le web qui fait de ce domaine le future pour les ingénieurs de développement.

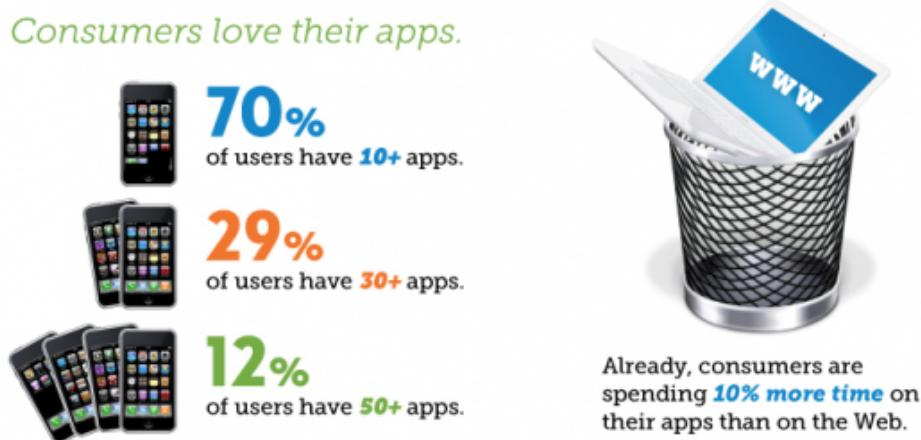


FIGURE 1.2 – Graphe de l'étude réalisé par Nuance Communications

Enfin, On a apprend les différents étapes pour créer une application mobile de Conception , Développement et Intégration (Testing) et ce concentrer surtout sur le dernier élément d'intégration en lisant les programmes publié par l'ISTQB [3].



FIGURE 1.3 – Logo d'ISTQB

### 1.3.2 Appliquer ces compétences dans un projet spécifié

Le projet que l'organisation m'a apporté c'est ProLab Mobile 5 et c'est une application mobile dédié pour gérer les fichiers des patients d'un laboratoire médicale et mon rôle dans ce projet c'est de l'étudier et faire des Test automatique pour vérifier que cette application fonctionne correctement. Et pour en avoir les acquis pour faire ces Tests la société m'a accordé quelque étapes. Tout d'abord, Je commence à lire et comprendre le premier article que l'organisation m'a donner et c'est le ISTQB19 [4] qui m'a apprend beaucoup de choses comme :

- Les différentes méthodes de Testing soit manuellement ou automatiquement via les simulateurs.
- Les fonctionnalités principale à ciblé dans un test.
- Apprendre à faire des Test Cases pour proprement divisé le travail de testing et avoir des résultats attendu pour bien évalué l'application.

En second lieu, J'ai reçu le deuxième article ISTQB18 [5] pour apprendre les test techniques "black-box" pour les implémenter dans le Case Testing de l'application.

Finalement, la société m'a demandé d'apprendre et maîtriser l'environnement Android Studio et le Framework Espresso [6] pour les appliquer dans un Test automatique.

## 1.4 Conclusion

Dans ce premier chapitre, nous avons présenté l'organisme d'accueil « Sama Consulting » . Après nous avons expliqué les différents tâches associés pendant le stage aussi que le but de ce stage et mon projet de faire une "Case Testing" pour l'application ProLab Mobile 5.

# Chapitre 2

## Préparation des Tests

### 2.1 Introduction

Ce chapitre est consacré à présenter la démarche suivie pour faire les Tests Case de l'application et les implémenter dans un environnement manuel.

### 2.2 Les Test Cases et le test manuelle

#### 2.2.1 Préparation des Test Cases

La premier tâche à faire c'est d'étudier les fonctionnalités principale de l'application et comprendre son rôle puis réaliser les "Test Case" pour cette application qui permet au Biogiste de valider et consulter les résultats d'une analyse médicale.

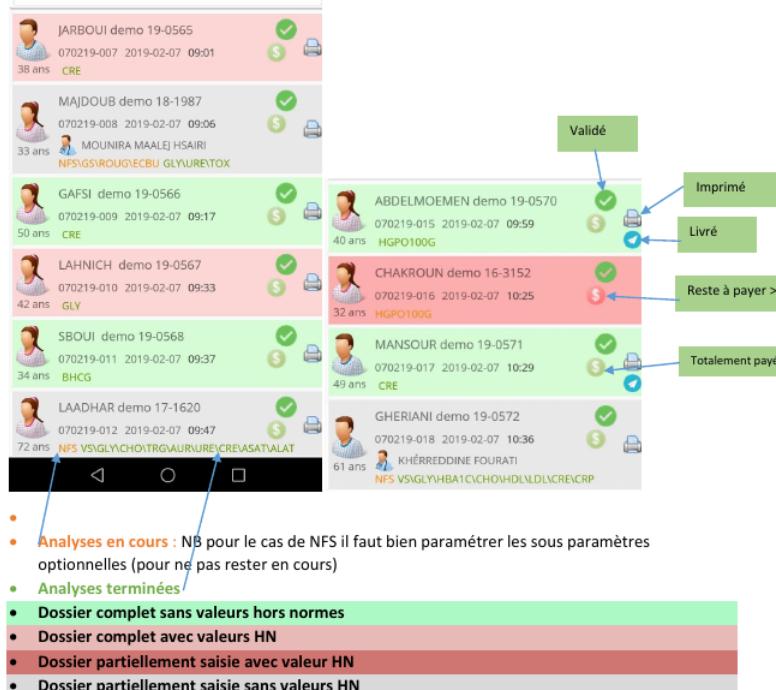


FIGURE 2.1 – Le Guide Code Couleurs de l'application ProLab Mobile 5

Ensuite, faut comprendre c'est quoi la structure d'une Test Case et les éléments nécessaires et ce sont :

- "Test Setup/Device" c'est où déroule le test soit dans une appareil mobile avec un test manuel ou soit dans une Emulator/Simulator avec un test automatique.
- "Test Case ID" et "Test Case Description" pour bien préciser le sujet qu'on va tester.
- Test Steps, c'est les étapes à suivre pour vérifier ce Test.
- "Pre-Conditions", c'est les conditions nécessaires qu'on est besoin d'avoir avant de faire ce Test.
- "Test Data", au cas où on est besoin de Data dans un Test et cette colonne peut-être affiché dans un fichier entier si la Data est trop volumineux.
- "Expected Result" et "Actual Result" sont les plus importantes parties d'un Test Case vu qu'on est besoin de connaître c'est quoi la résultat attendu et si elle matche le résultat données.
- "Status" Si le résultat attendu et même que le résultat donné alors c'est une test succès sinon c'est un échec.
- "Executed By" c'est celui qui a fait/écrit le Test.
- "Executed Date" est la date d'exécution de cette Test.

Project	Prolab-Mobile-S	Module	Login functionnalities	Created By		Creation Date		Reviewed	Mohamed Azz Karoui	Reviewed Date	14/08/2020
Test	Test Scenario	Test	Test Case	Test Case Description	Test Steps	Pre-conditions	Test Data	Expected Results	Actual Results	Screenshots	Status
TS_PMS_1	Verify the Login functionality in different modes	Galaxy J2 / Android 5.1.1	TC_PMS_1_1	Use the application in an Offline mode	1- Turn off any internet source on the mobile 2- Open the application 3- Click the "Se Conneter"	Valid Login Settings Mobile in Offline mode		A popup message box to show to tell us to verify our Internet	A popup message that says "Veuillez vérifier votre connexion internet"	Screenshot	Pass
		Galaxy J2 / Android 5.1.1	TC_PMS_1_2	Use the application with internet	1- Turn on any internet source on the mobile 2- Open the application	Valid Login Settings Mobile in Online mode		The application will automatically log you in	A popup message that says "Connexion en cours... " that transition automatically later to the application	Screenshot	Pass
		Galaxy J2 / Android 5.1.1	TC_PMS_1_3	Use the application in a Flight mode	1- Turn on the Application mode on 2-Open the application	Valid Login Settings Mobile in Airplane		A popup message box to show to tell us to verify our Internet	A popup message that says "Veuillez vérifier votre connexion internet"	Screenshot	Pass
TS_PMS_6	Verify the Login Settings	Galaxy J2 / Android 5.1.1	TC_PMS_6_1	Verify the app functionality with a wrong server address	1- Open the application 2- Click on the 3 points top right 3- Click on Settings 4- Click on General 5- Click on Server Address 6- Enter an invalid server address	Mobile have access to Internet	Server Address : 197.240.68.19	A popup message to show that there's an error in the login Settings	A popup message to show that there's an error in the login Settings	Screenshot 1	Pass
		Galaxy J2 / Android 5.1.1	TC_PMS_6_2	Verify the app functionality with any wrong login setting	1- Open the application 2- Click on the 3 points top right 3- Click on Settings 4- Click on General 5- Click on any setting and put an invalid data instead	Mobile have access to Internet	Port number: 7272 Version:Prolab-Prolab9 User name : user	A popup message to show that there's an error in the login Settings	A popup message to show that there's an error in the login Settings	Screenshot 2	Pass

FIGURE 2.2 – Les Premiers Test Cases effectués

Test Scenario ID	Test Scenario Description	Test Setup/Device	Test Case ID	Test Case Description	Test Steps	Pre-conditions	Test Data	Expected Results	Actual Results	Screenshots	Status
TS_PMS_2	Verify the Date functionality	Galaxy J2 / Android 5.1.1	TC_PMS_2.1	Enter an Invalid "From date" and a Valid "To date"	1- Enter an Invalid "From Date" 2- Enter a Valid "To Date"	Valid Login Settings Mobile have access to Internet	From Date : "1/1/1900" To Date : "11/08/2020"	A popup message that signals error about the "From Date" being invalid	A popup message that tells you how many cases you have in that set period of time and displays them	Screenshot	Fail
		Galaxy J2 / Android 5.1.1	TC_PMS_2.2	Enter a Valid "From date" and an Invalid "To date"	1- Enter a Valid "From Date" 2- Enter an Invalid "To Date"	Valid Login Settings Mobile have access to Internet	From Date : "11/08/2020" To Date : "31/12/2100"	A popup message that signals error about the To Date" being invalid	A popup message that tells you how many cases you have in that set period of time and displays	Screenshot	Fail
		Galaxy J2 / Android 5.1.1	TC_PMS_2.3	Enter a Valid "From date" and a Valid "To date"	1- Enter a Valid "From Date" 2- Enter a Valid "To Date"	Valid Login Settings Mobile have access to Internet	From Date : "01/01/2010" To Date : "11/08/2020"	An update that displays all the cases in the between these 2 dates	A popup message that tells you how many cases you have in that set period of time and displays	Screenshot	Pass
TS_PMS_3	Verify the Search functionality	Galaxy J2 / Android 5.1.1	TC_PMS_3.1	Enter a Valid patient ID in the Search function	1- Click on the Search button 2- Enter a Valid Patient ID	Valid Login Settings Mobile have access to Internet	Patient ID : "Patient 16-3102"	An update that displays the patient with the ID "PATIENT 16-3102"	An update that displays the patient with the ID "PATIENT 16-3102"	Screenshot	Pass
		Galaxy J2 / Android 5.1.1	TC_PMS_3.2	Enter a Valid date in the Search function	1- Click on the Search button 2- Enter a valid date with the format ddmmyy	Valid Login Settings Mobile have access to Internet	Date : "290916"	An update that displays all the patient that's been registered on that day	An update that displays all the patient that's been registered on that day	Screenshot	Pass
		Galaxy J2 / Android 5.1.1	TC_PMS_3.3	Enter an Invalid patient ID in the Search function	1- Click on the Search button 2- Enter an invalid Patient ID	Valid Login Settings Mobile have access to Internet	Patient ID : "Patient 16-3061"	An update that displays that there's no patients with such ID	An update that shows there's 0 files with such ID	Screenshot	Pass
		Galaxy J2 / Android 5.1.1	TC_PMS_3.4	Enter an invalid date in the Search function	1- Click on the Search button 2- Enter an invalid date with the format ddmmvv	Valid Login Settings Mobile have access to Internet	Date : "290990"	An update that displays that there's no patients registered in that date	An update that shows there's no files registered in that date	Screenshot	Pass

FIGURE 2.3 – Les Deuxièmes Test Cases effectués

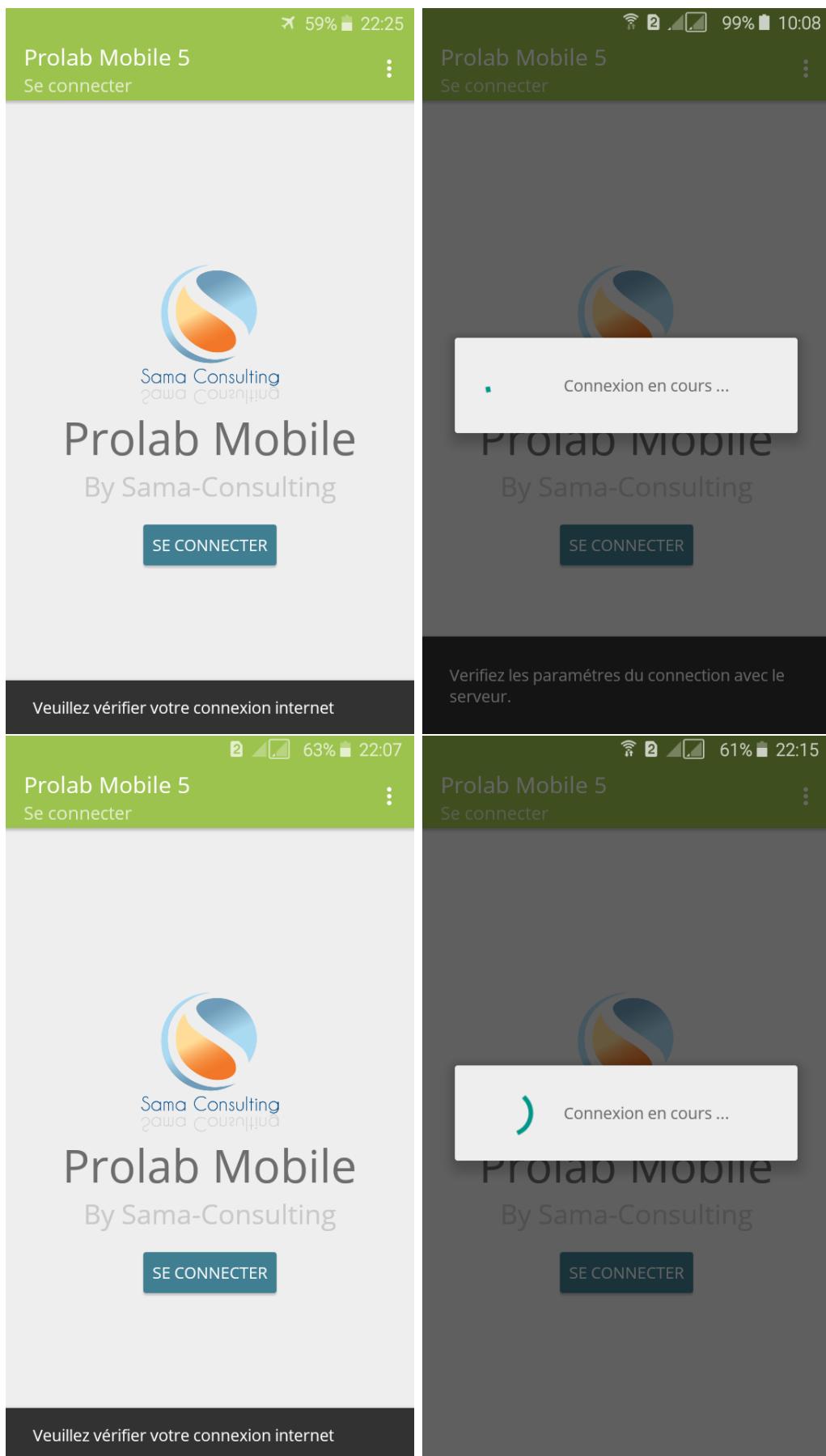
TS_PMS_4	Verify the ability to Phonecall a Patient	Galaxy J2 / Android 5.1.1	TC_PMS_4.1	Choose a Patient that has an invalid or no Phone number	1- Click on a patient's file 2- Click on the phone icon	Valid Login Settings Mobile have access to Internet		A popup to signals an error or that the number doesn't exist	A popup that signals that contact is not available	Screenshot	Pass
		Galaxy J2 / Android 5.1.1	TC_PMS_4.2	Choose a Patient that has a Valid Phone number	1- Click on a patient's file 2- Click on the phone icon	Valid Login Settings Mobile have access to Internet		A phone call starts	A phone call starts	Screenshot 1	Pass
		Galaxy J2 / Android 5.1.1	TC_PMS_5.1	Verify the ability to validate a Patient's file	1- Click on a patient's file that's currently unvalidated 2- Click on the tick icon	Valid Login Settings Mobile have access to Internet Unvalidated File		File is validated	File is validated	Screenshot 1	Pass
		Galaxy J2 / Android 5.1.1	TC_PMS_5.2	Verify the ability to invalidate a Patient's file	1- Click on a patient's file that's currently validated 2- Click on the tick icon	Valid Login Settings Mobile have access to Internet Validated File		File is unvalidated	File is unvalidated	Screenshot 1	Pass

FIGURE 2.4 – Les Troisièmes Test Cases effectués

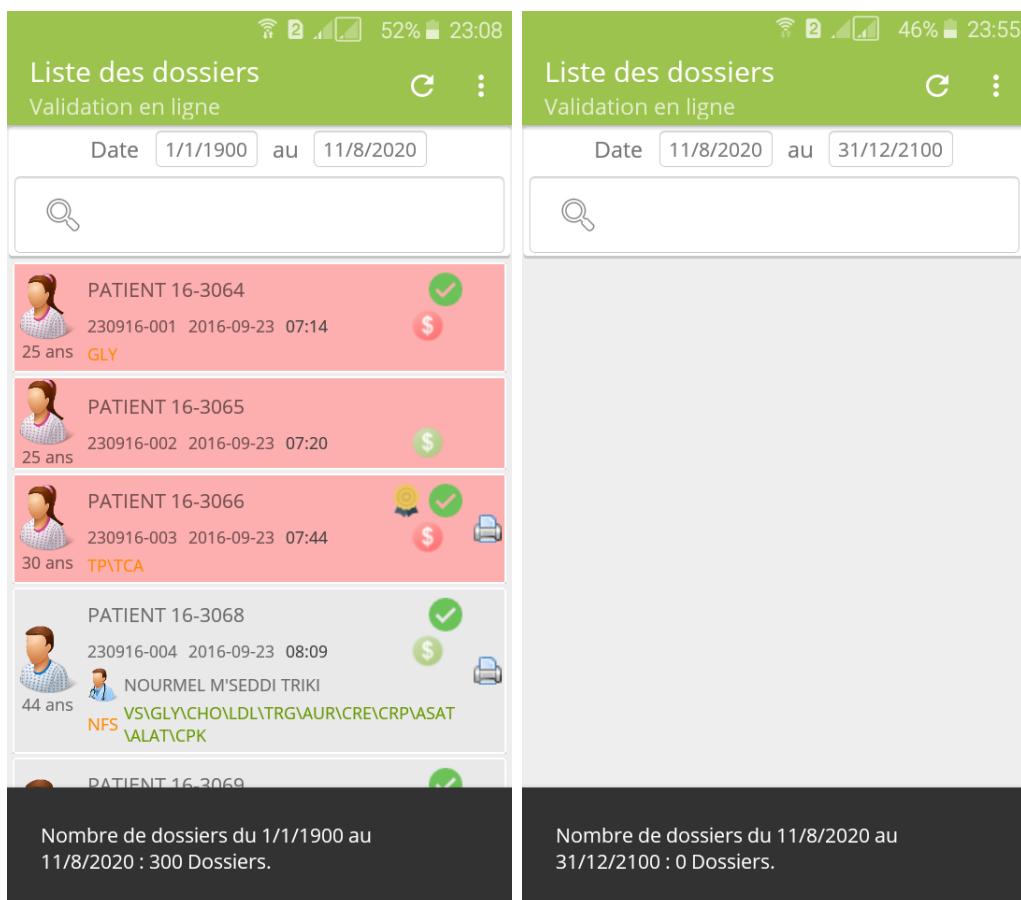
## 2.2.2 Appliquer sur un Environnement manuel

Enfin pour voir si la résultat attendu et la même que celle donnée , J'ai utilisé mon téléphone Galaxy J2 / Android 5.1.1 pour effectué ces tests que j'ai préparé manuellement et voici les résultats de ces test :

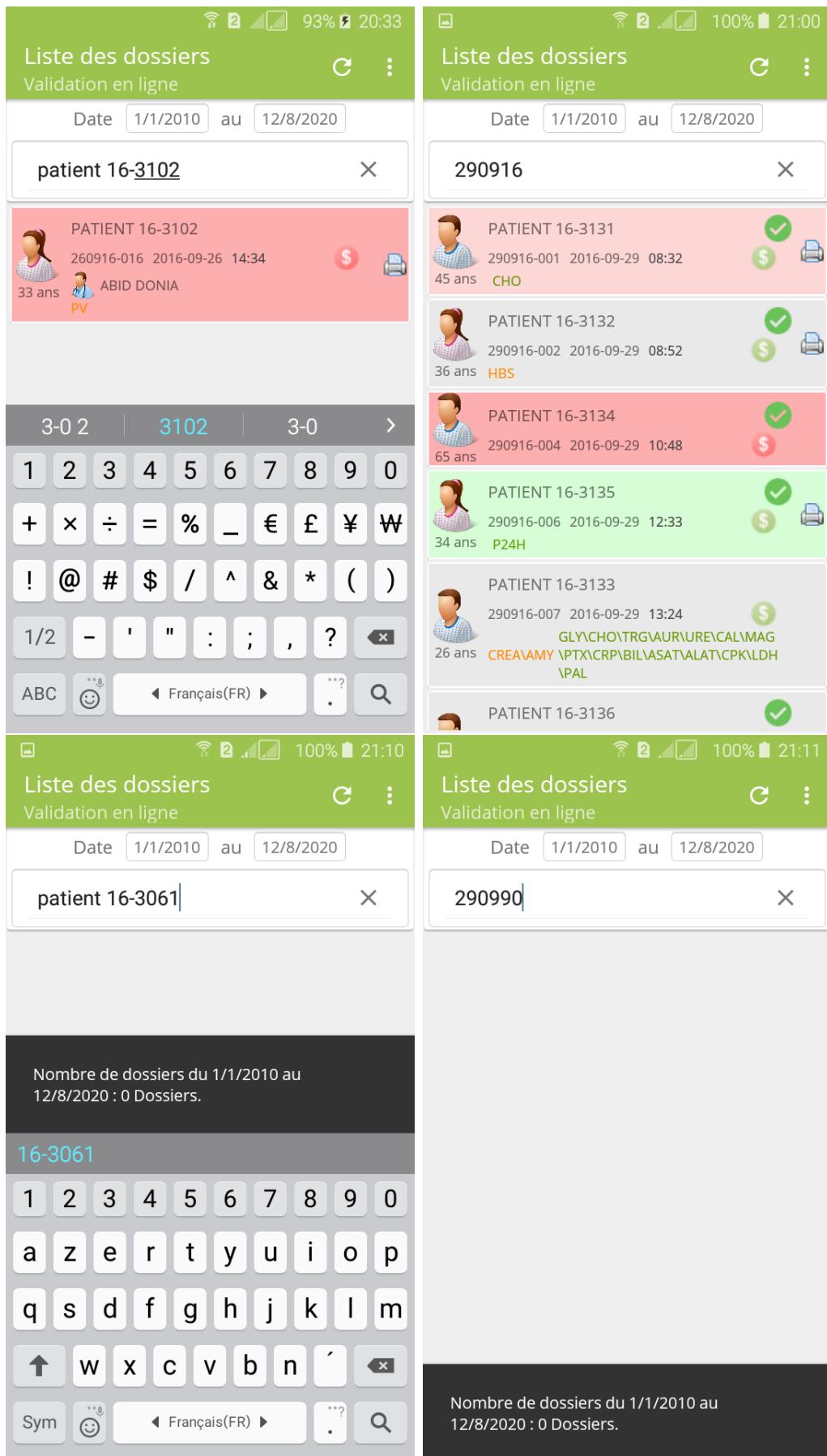
Vérifier les fonctionnalité de Login dans différentes modes de connectivité



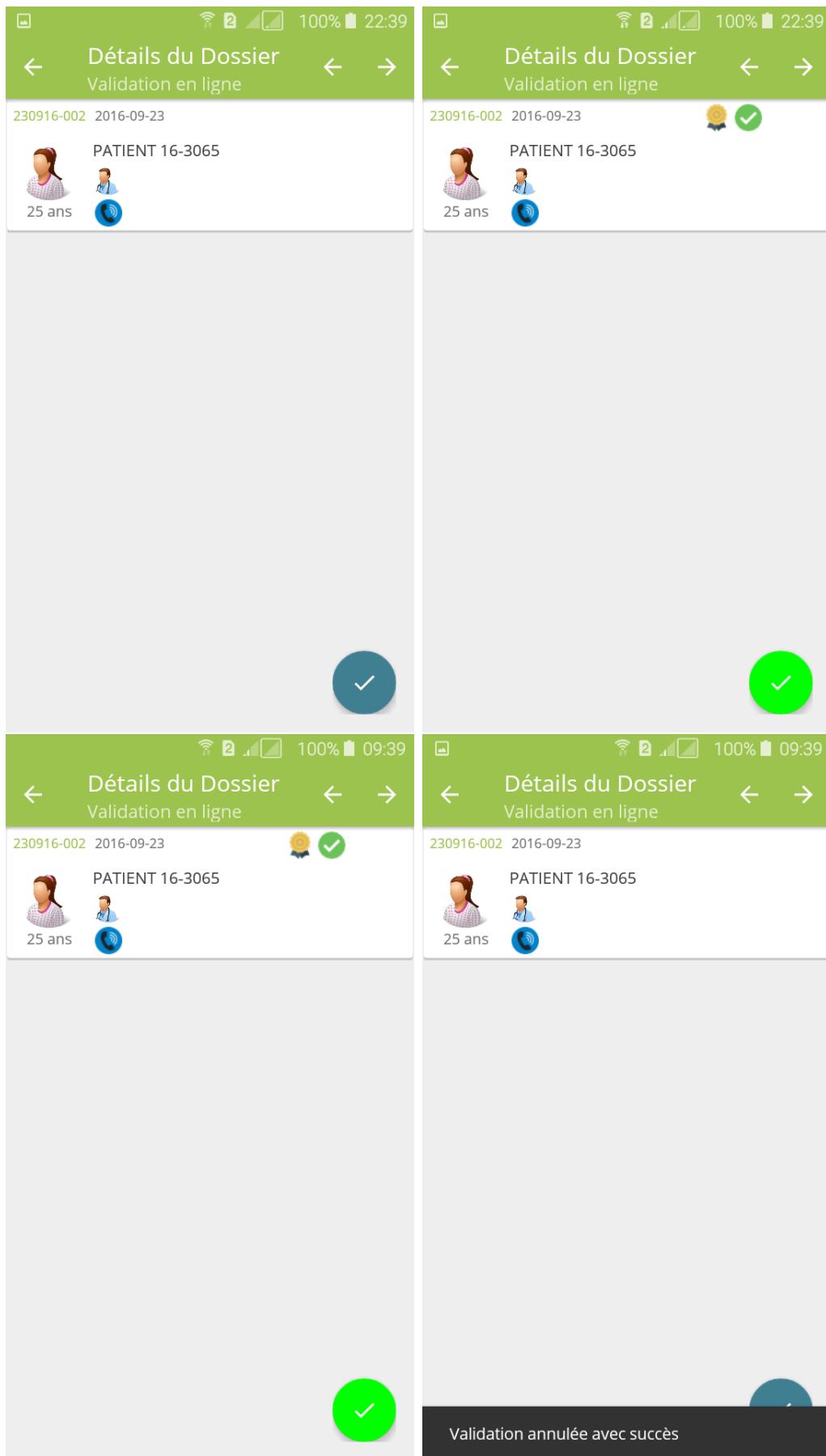
## Vérifier la fonctionnalité Date



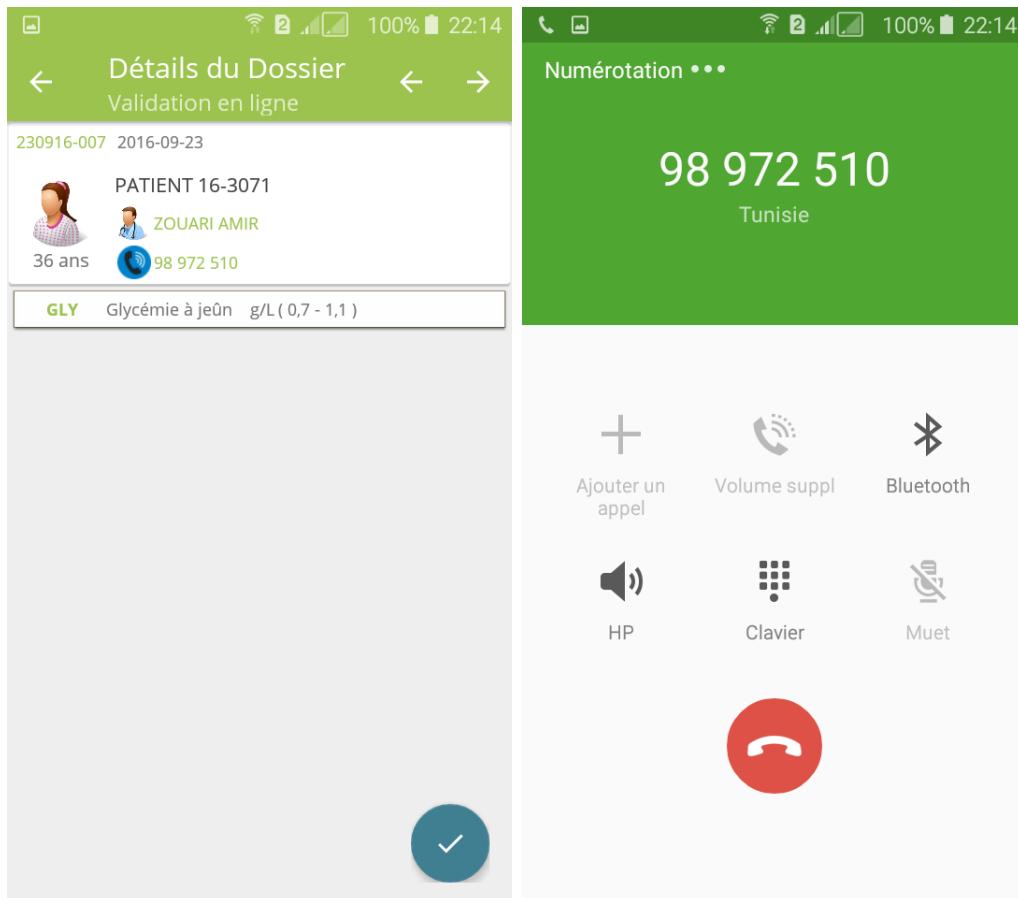
## Vérifier la fonctionnalité du Chercher un patient



Vérifier la fonctionnalité de valider et invalider un dossier d'un patient



Vérifier la fonctionnalité d'appelé un patient



## 2.3 Conclusion

Dans ce deuxième chapitre, nous avons étudier l'application ProLab Mobile 5, présenté la structure de Test Case puis nous avons présenté les tâches effectuées manuellement.

# Chapitre 3

## L'automation du Test

### 3.1 Introduction

Ce chapitre est consacré à présenter le démarche suivi pour faire l'automation de ces Test Cases préparé dans le chapitre précédent.

### 3.2 Implémenter Android Studio et Espresso

Pour faire l'automation de ces Test Cases nous allons Installer et préparer l'environnement Android Studio et apprendre son Framework Espresso.

#### 3.2.1 Définition d'Espresso

Espresso est un Framework de test pour Android permettant d'écrire facilement des tests d'interface utilisateur fiables.

Google a publié le framework Espresso en octobre 2013. Depuis sa version 2.0, Espresso fait partie du référentiel de support Android.

Espresso synchronise automatiquement vos actions de test avec l'interface utilisateur de votre application. Le framework peut aussi garantir que votre activité est démarrée avant l'exécution des tests. Il laisse également le test attendre la fin de toutes les activités de fond observées[7].

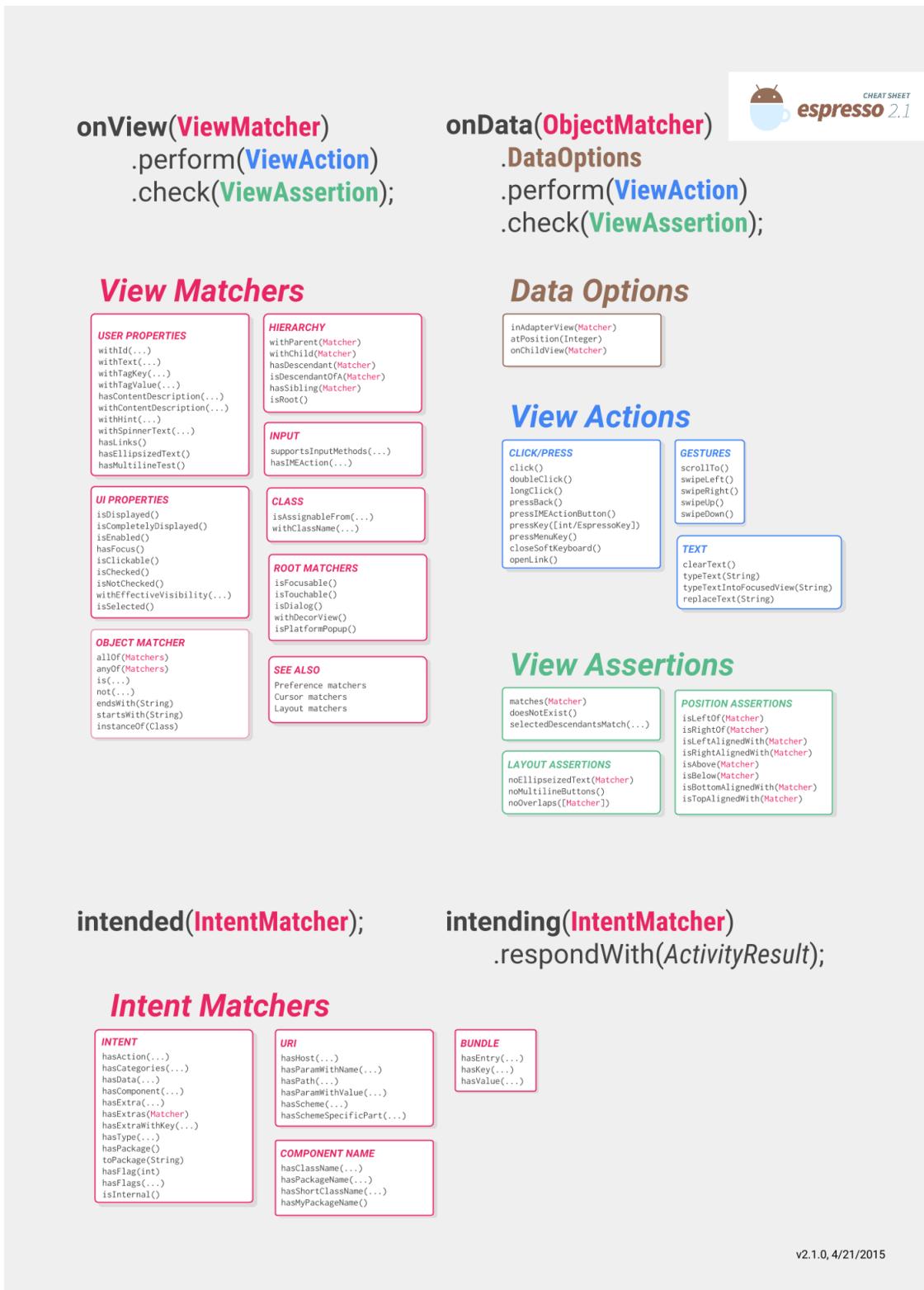


FIGURE 3.1 – Logo du Framework Espresso

Enfin l'une des plus important qualité d'Espresso c'est qu'il est facile à manipuler , facile à installé et surtout beaucoup plus stable que d'autre Framework comme « Appium» [8].

		
<b>Owner</b>	Open source	Google/open source
<b>Scope</b>	Cross platform	Android
<b>App type</b>	Native, hybrid, web	Native, hybrid
<b>Languages</b>	Java, C#, Ruby, Python	Java/Kotlin
<b>Source required?</b>	No	Yes
<b>Out of app</b>	Android - yes iOS - No	No
<b>Speed</b>	Slow	Fast
<b>Setup</b>	Hard	Easy
<b>Usage</b>	Hard (previous selenium knowledge helps)	Easy for dev, Medium for SDET
<b>Stability</b>	Low	High
<b>CI integration</b>	Hard	Medium

FIGURE 3.2 – Comparaison entre Espresso et Appium



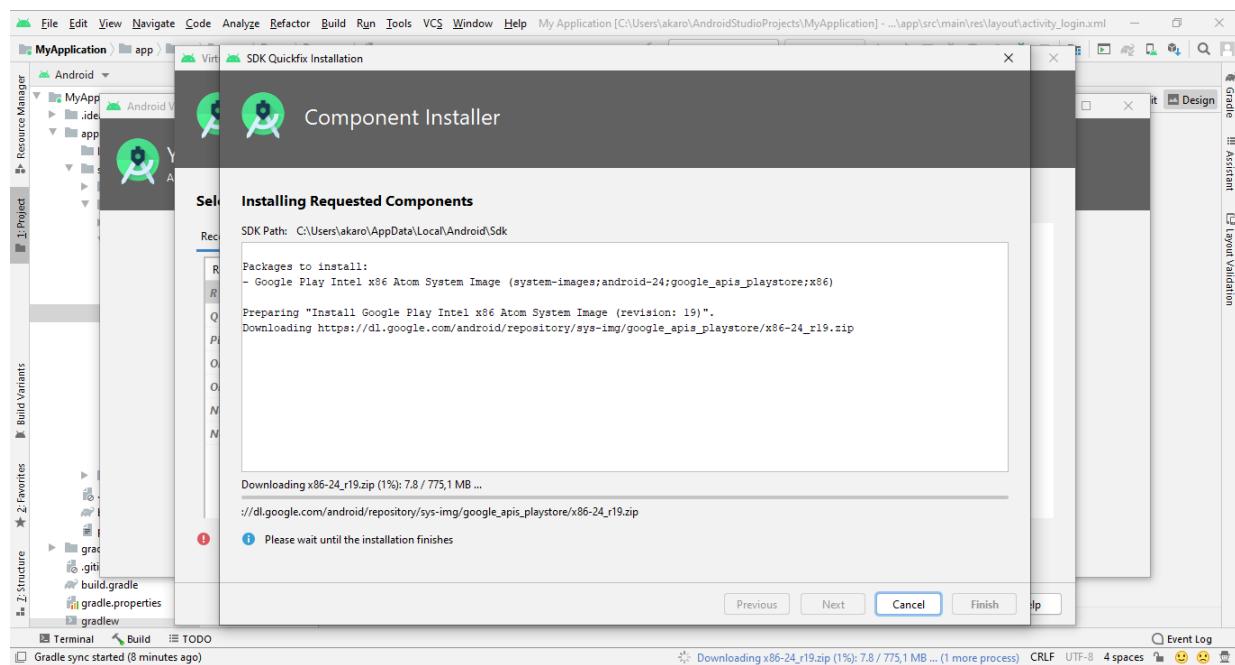
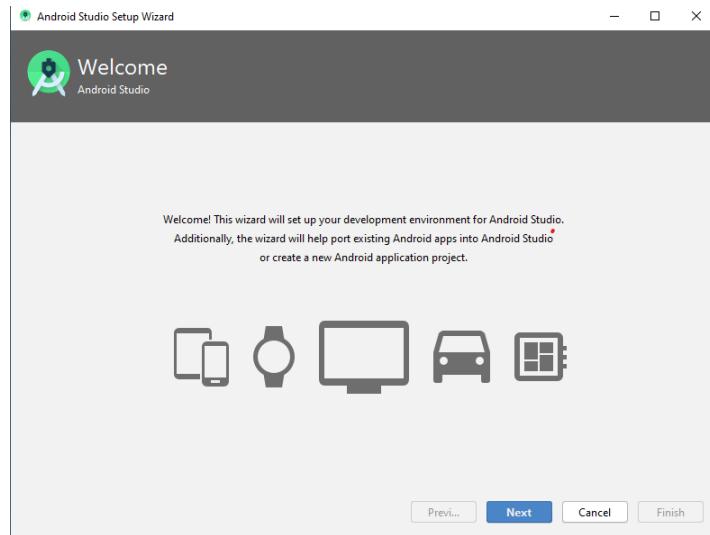
The Espresso Cheat Sheet v2.1 is a comprehensive guide to the Espresso testing framework for Android. It is organized into several sections:

- onView(ViewMatcher)**: A code snippet showing how to perform actions and check assertions on a view.
- onData(ObjectMatcher)**: A code snippet showing how to perform actions and check assertions on data.
- View Matchers**: A section detailing various matchers for views, including:
  - USER PROPERTIES**: Methods like `withId(...)`, `withText(...)`, etc.
  - HIERARCHY**: Methods like `withParent(Matcher)`, `withChild(Matcher)`, etc.
  - INPUT**: Methods like `supportsInputMethods(...)`, `hasIMEAction(...)`.
  - CLASS**: Methods like `isAssignableFrom(...)`, `withClassName(...)`.
  - ROOT MATCHERS**: Methods like `isFocusable()`, `isTouchable()`, `isDialog()`, etc.
  - SEE ALSO**: References to Cursor matchers and Layout matchers.
- Data Options**: A section detailing data options for onData, including methods like `inAdapterView(Matcher)`, `atPosition(Integer)`, and `onChildView(Matcher)`.
- View Actions**: A section detailing various actions for views, categorized into Click/Press and Gestures:
  - CLICK/PRESS**: Methods like `click()`, `doubleClick()`, `longClick()`, etc.
  - GESTURES**: Methods like `scrollTo()`, `swipeLeft()`, `swipeRight()`, etc.
- View Assertions**: A section detailing various assertions for views, including Position Assertions and Layout Assertions:
  - POSITION ASSERTIONS**: Methods like `isLeftOf(Matcher)`, `isRightOf(Matcher)`, etc.
  - LAYOUT ASSERTIONS**: Methods like `noEllipsizeText(Matcher)`, `noMultilineButtons()`, etc.
- intended(IntentMatcher)**: A code snippet showing how to intended an intent matcher.
- intending(IntentMatcher)**: A code snippet showing how to intending an intent matcher and respond with an activity result.
- Intent Matchers**: A section detailing various matchers for intents, including:
  - INTENT**: Methods like `hasAction(...)`, `hasCategories(...)`, etc.
  - URI**: Methods like `hasHost(...)`, `hasParamWith Name(...)`, etc.
  - BUNDLE**: Methods like `hasEntry(...)`, `hasKey(...)`, etc.
  - COMPONENT NAME**: Methods like `hasClassName(...)`, `hasPackageName(...)`, etc.

At the bottom right, the version is indicated as v2.1.0, 4/21/2015.

FIGURE 3.3 – Espresso Cheatsheet

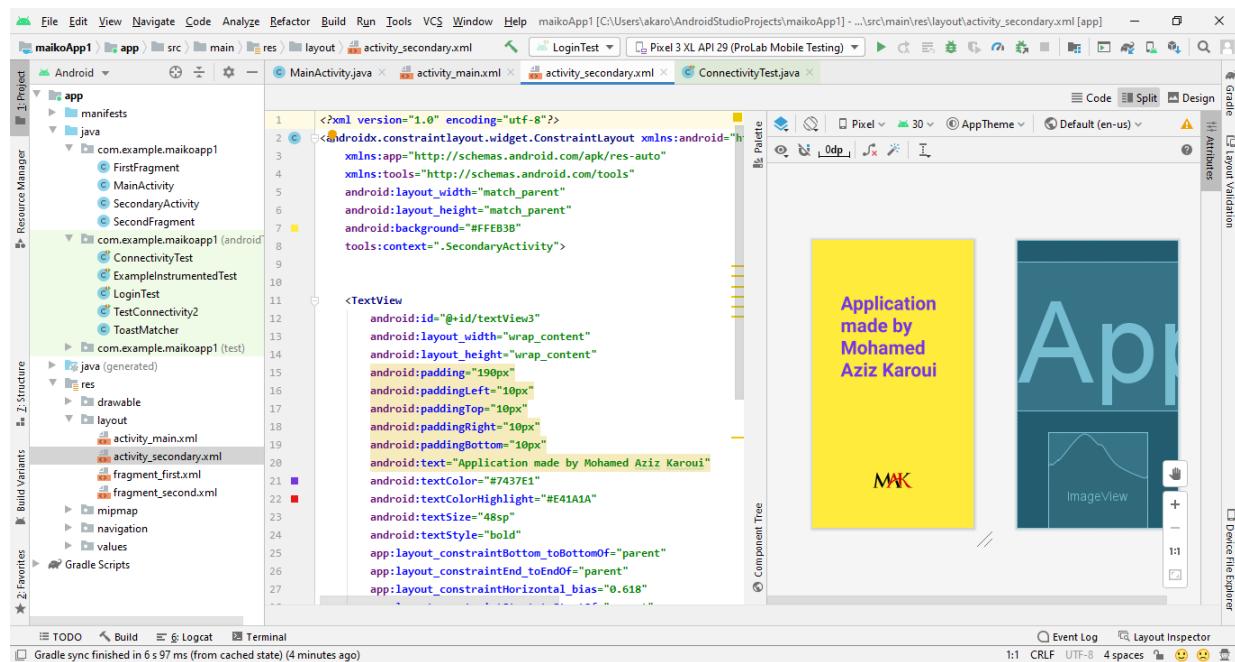
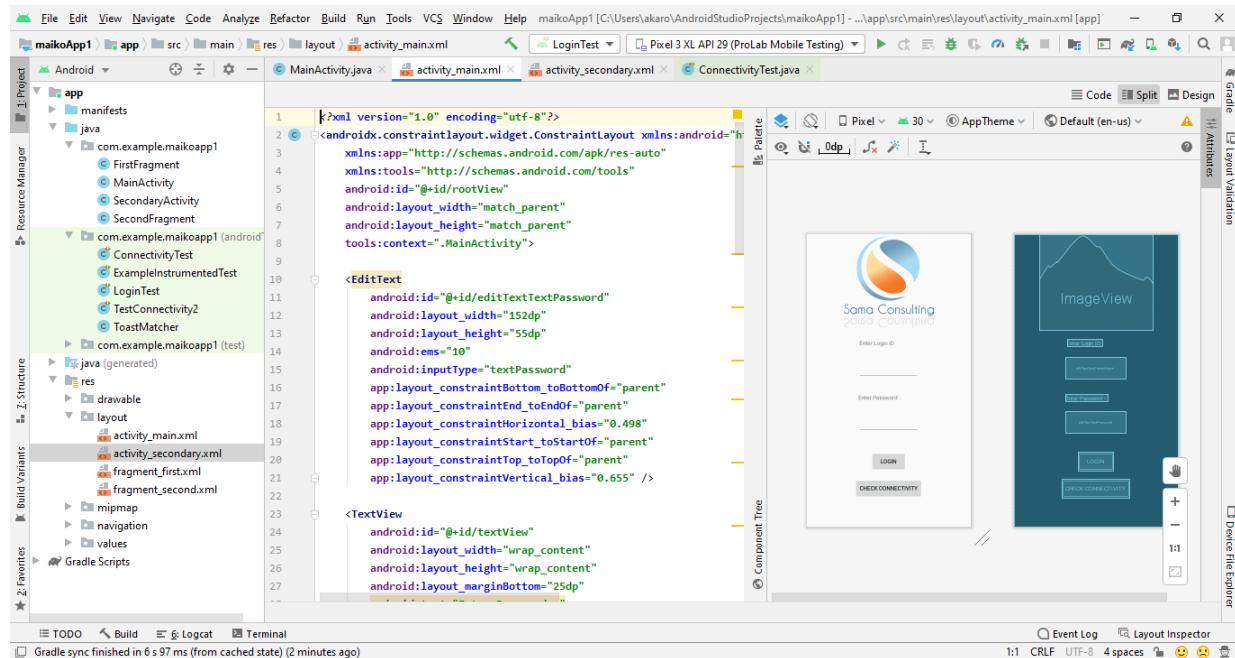
### 3.2.2 Installation de l'environnement Android Studio et son Framework Espresso

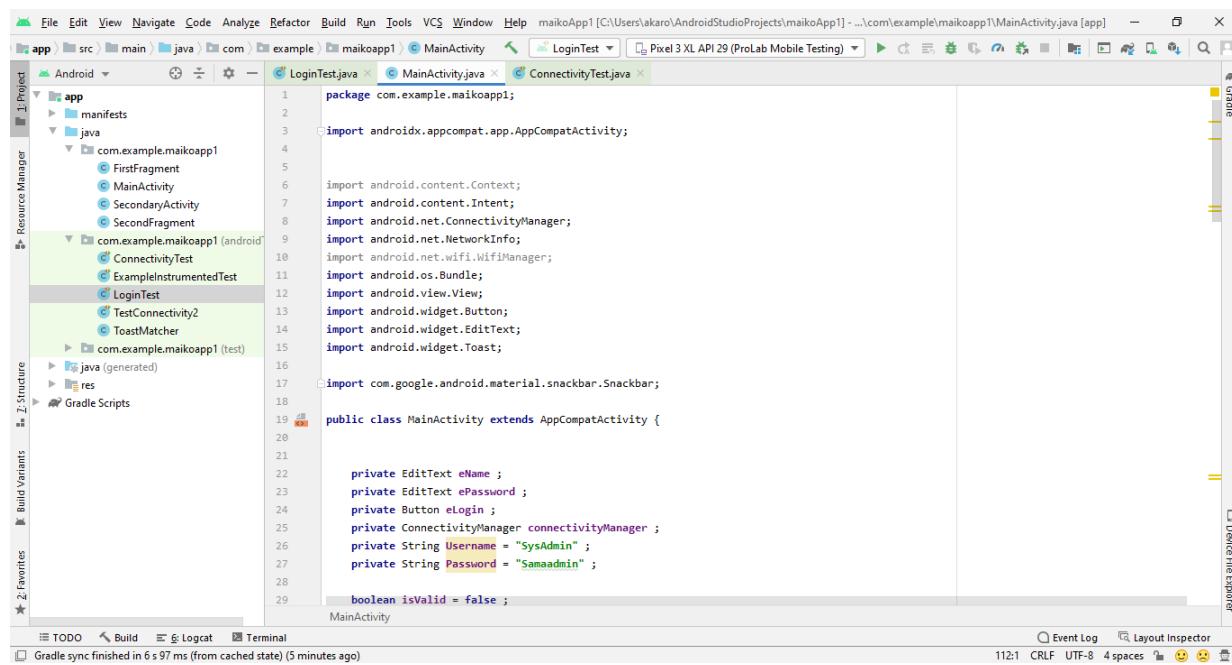


### 3.3 Crée une Application de Test

On va créer une application de Test qui a le rôle du simple Login et on va essayer quelques Espresso Tests pour vérifier que l'environnement est bien installé et qu'on maîtrise ce Framework Espresso.

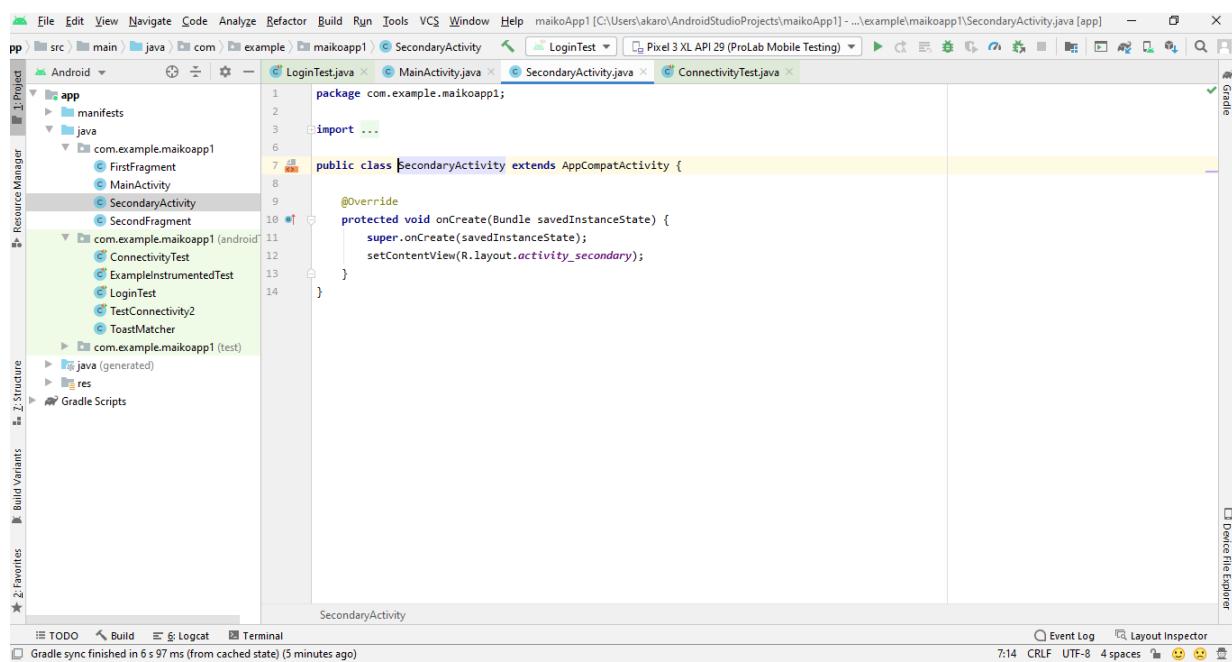
Et on va commencer par préparer cette application





The screenshot shows the Android Studio interface with the following details:

- Project Structure:** The project is named "maikoApp1". The "app" module contains "manifests", "java", and "com.example.maikoapp1" packages. The "com.example.maikoapp1" package contains "FirstFragment", "MainActivity", "SecondaryActivity", and "SecondFragment" classes, along with "com.example.maikoapp1 (android)" and "com.example.maikoapp1 (test)" sub-packages. The "com.example.maikoapp1 (test)" package contains "ConnectivityTest", "ExampleInstrumentedTest", "LoginTest", "TestConnectivity2", and "ToastMatcher" classes.
- MainActivity.java:** The code defines a class `MainActivity` that extends `AppCompatActivity`. It includes imports for Context, Intent, ConnectivityManager, NetworkInfo, WifiManager, Bundle, View, Button, EditText, and Toast. It also imports com.google.android.material.snackbar.Snackbar. The class has private fields for EditTexts `eName` and `ePassword`, a Button `elogin`, and a ConnectivityManager `connectivityManager`. It has constants for `Username` ("Sysadmin") and `Password` ("Samaadmin"). A boolean `isValid` is set to `false`.
- Code Editor:** The code editor shows the Java code for `MainActivity`. The line `super.onCreate(savedInstanceState);` is highlighted in yellow.
- Toolbars and Status Bar:** The top bar shows standard Android Studio menu items like File, Edit, View, Navigate, Code, Analyze, Refactor, Build, Run, Tools, VCS, Window, Help. The status bar at the bottom shows "Gradle sync finished in 6 s 97 ms (from cached state) (5 minutes ago)", the time "11:21", and encoding "CRLF UTF-8 4 spaces".



The screenshot shows the Android Studio interface with the following details:

- Project Structure:** The project is named "maikoApp1". The "app" module contains "manifests", "java", and "com.example.maikoapp1" packages. The "com.example.maikoapp1" package contains "FirstFragment", "MainActivity", "SecondaryActivity", and "SecondFragment" classes, along with "com.example.maikoapp1 (android)" and "com.example.maikoapp1 (test)" sub-packages. The "com.example.maikoapp1 (test)" package contains "ConnectivityTest", "ExampleInstrumentedTest", "LoginTest", "TestConnectivity2", and "ToastMatcher" classes.
- SecondaryActivity.java:** The code defines a class `SecondaryActivity` that extends `AppCompatActivity`. It includes imports for ... and AppCompatActivity. The class has an override method `onCreate` that calls `super.onCreate(savedInstanceState)` and sets the content view to `R.layout.activity\_secondary`.
- Code Editor:** The code editor shows the Java code for `SecondaryActivity`. The line `super.onCreate(savedInstanceState);` is highlighted in yellow.
- Toolbars and Status Bar:** The top bar shows standard Android Studio menu items like File, Edit, View, Navigate, Code, Analyze, Refactor, Build, Run, Tools, VCS, Window, Help. The status bar at the bottom shows "Gradle sync finished in 6 s 97 ms (from cached state) (5 minutes ago)", the time "7:14", and encoding "CRLF UTF-8 4 spaces".

The screenshot shows the Android Studio interface with the project 'maikoApp1' open. The left sidebar displays the project structure under 'app'. The code editor shows the file 'LoginTest.java' which contains Java code for testing the application's login functionality using Espresso. The code includes imports for JUnit and Espresso, annotations like @RunWith and @LargeTest, and a test method 'Test1ValidLogin'.

```

import org.junit.runner.RunWith;
import static androidx.test.espresso.Espresso.onView;
import static androidx.test.espresso.action.ViewActions.click;
import static androidx.test.espresso.action.ViewActions.closeSoftKeyboard;
import static androidx.test.espresso.action.ViewActions.scrollTo;
import static androidx.test.espresso.action.ViewActions.typeText;
import static androidx.test.espresso.action.ViewActions.typeTextIntoFocusedView;
import static androidx.test.espresso.matcher.ViewMatchers.isDisplayed;
import static androidx.test.espresso.matcher.ViewMatchers.withId;
import static org.junit.Assert.*;
import org.junit.FixMethodOrder;
import org.junit.runners.MethodSorters;

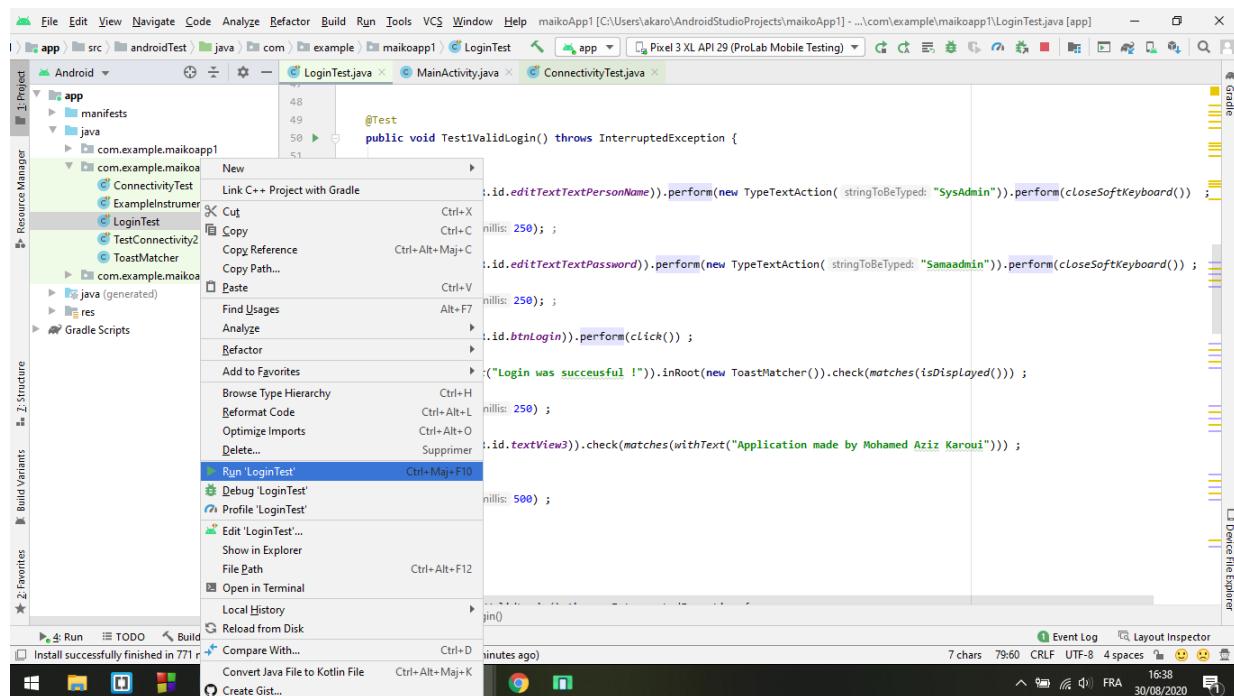
@RunWith(AndroidJUnit4.class)
@LargeTest
@FixMethodOrder(MethodSorters.NAME_ASCENDING)
public class LoginTest {

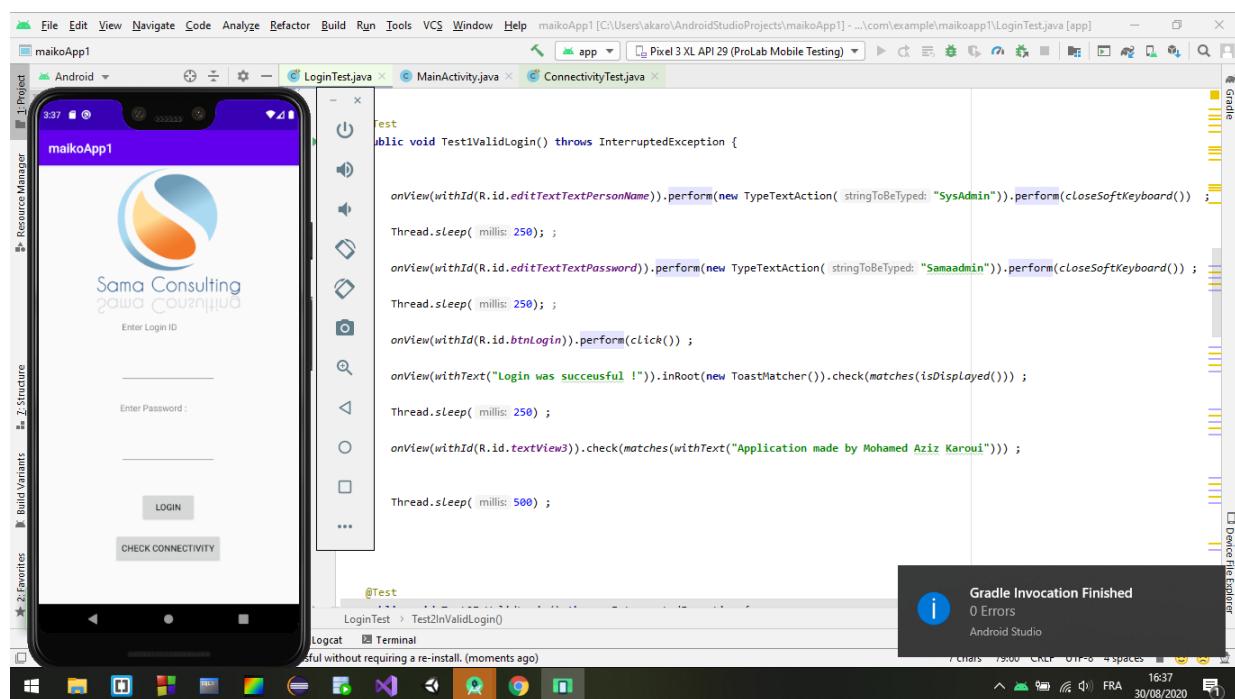
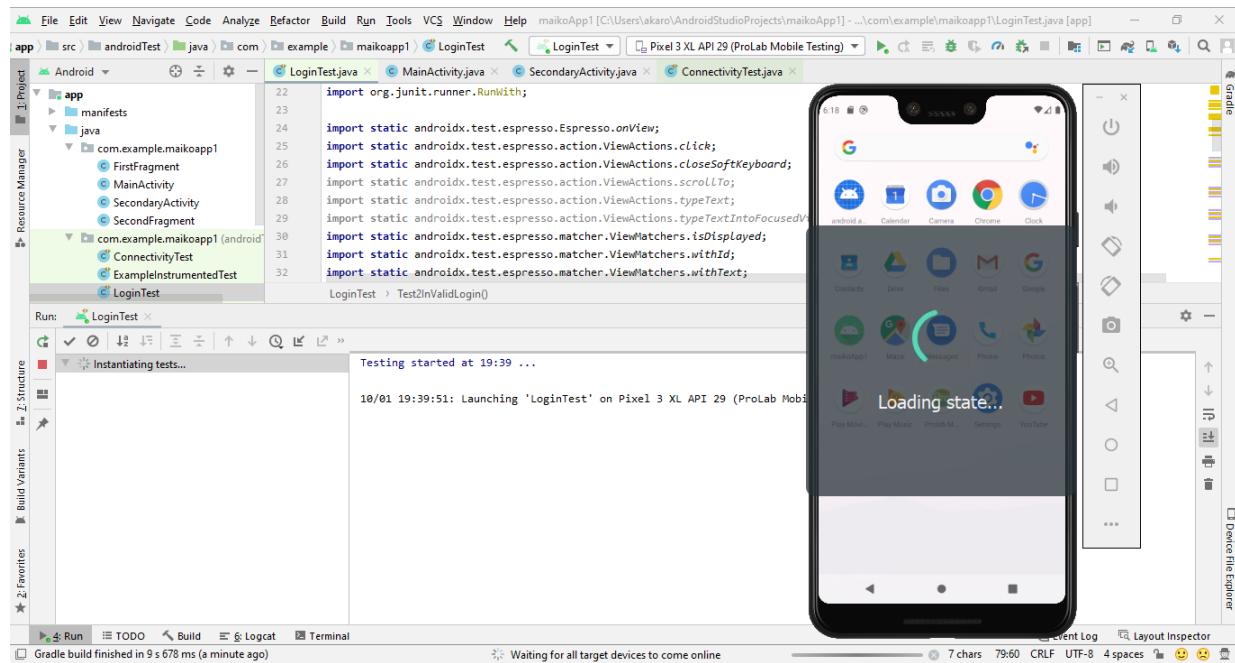
    @Rule
    public ActivityTestRule<MainActivity> activityRule =
            new ActivityTestRule<>(MainActivity.class);

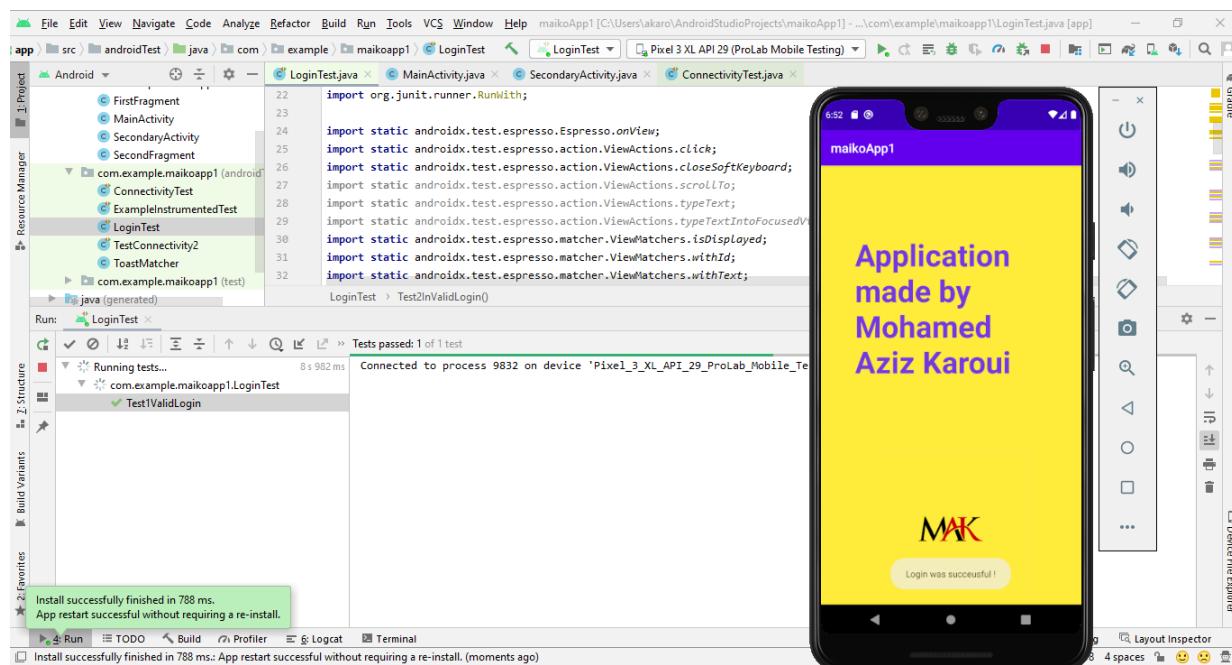
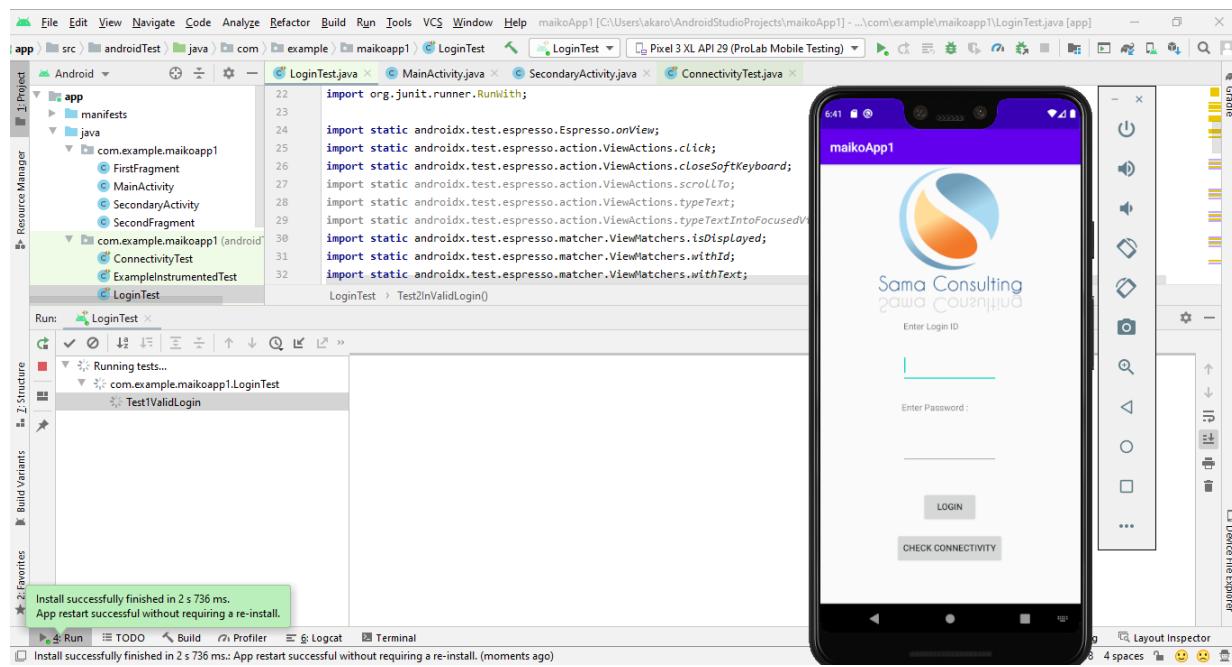
    @Test
    public void Test1ValidLogin() throws InterruptedException {
        ...
    }
}

```

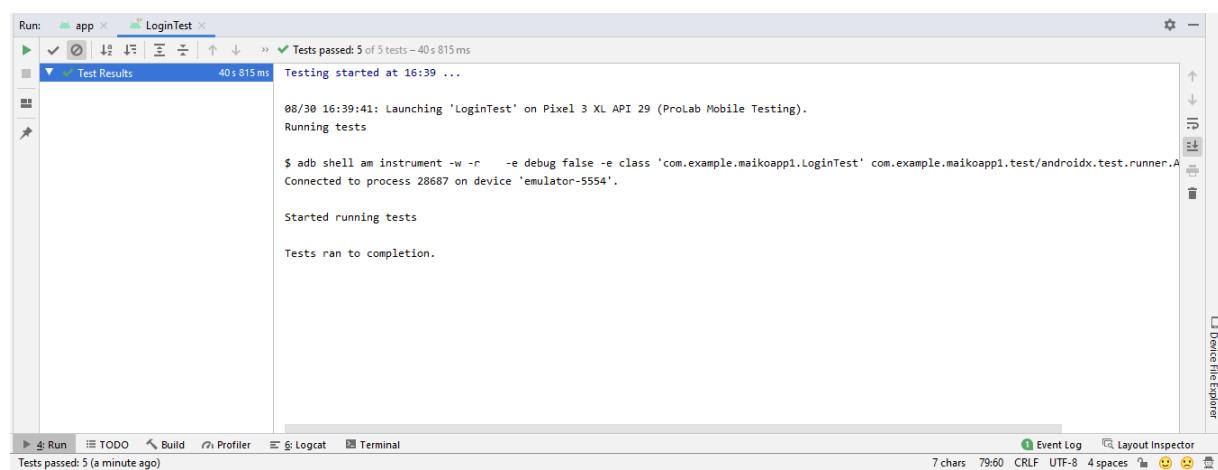
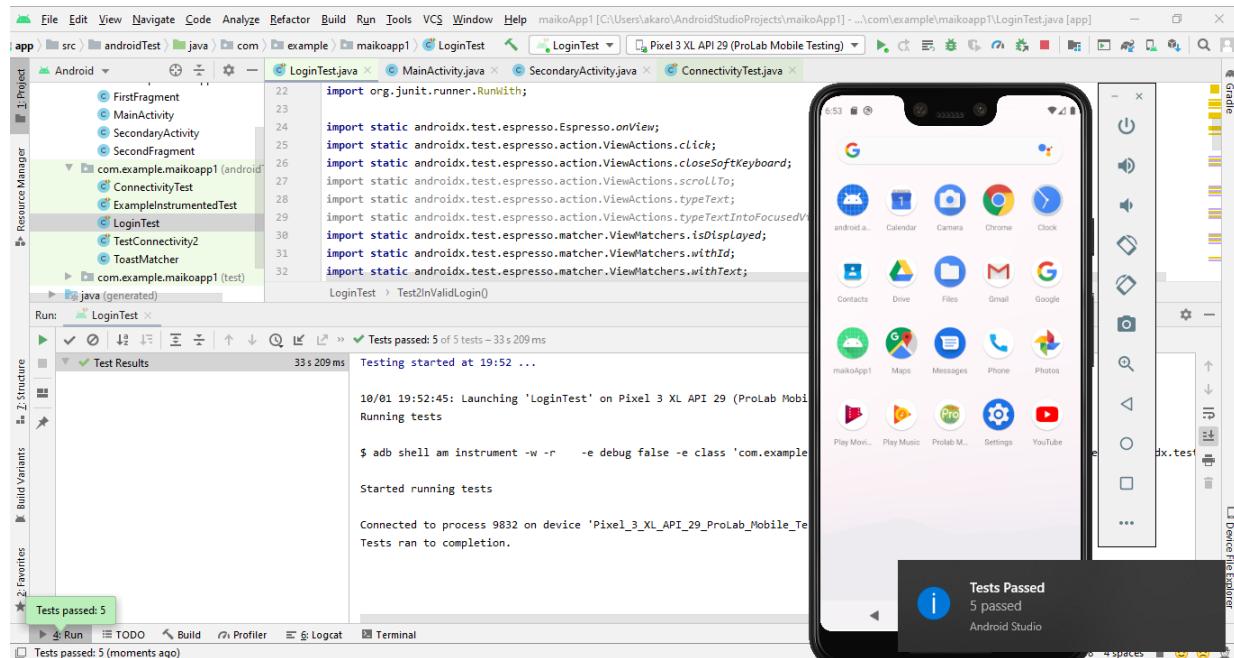
Puis on prépare les tests pour vérifier les simples étapes de Login et voir si l'application répond correctement dans tout les situations.







Et enfin on aura la résultat que le test est réussit



## 3.4 Faire des Test automatique sur ProLab Mobile 5

Enfin on va traduire les Test Case qu'on a fait au début de notre stage pour le test manuel dans un code Espresso avec l'aide de Espresso Record Test pour vérifier que notre application est testé automatiquement.

```

208
209
210
211
212
213
214
215
216
217
218
219
220
221
222
223
224
225 @
VerifyLoginSettingsWrong > verifyLoginSettingsCorrect()
    ViewInteraction textView = onView(
        allOf(withId("com.sama_consulting.prolabmobile5:id/textView"),
            childAtPosition(
                allOf(withId(R.id.action_bar),
                    childAtPosition(
                       (withId(R.id.action_bar_container),
                            position: 0)),
                    position: 1),
                isDisplayed())));
    textView.check(matches(withText("Se connecter")));
    Thread.sleep( millis: 500 );
}
private static Matcher<View> childAtPosition(
    final Matcher<View> parentMatcher) {
    return childAtPosition(parentMatcher, 0);
}

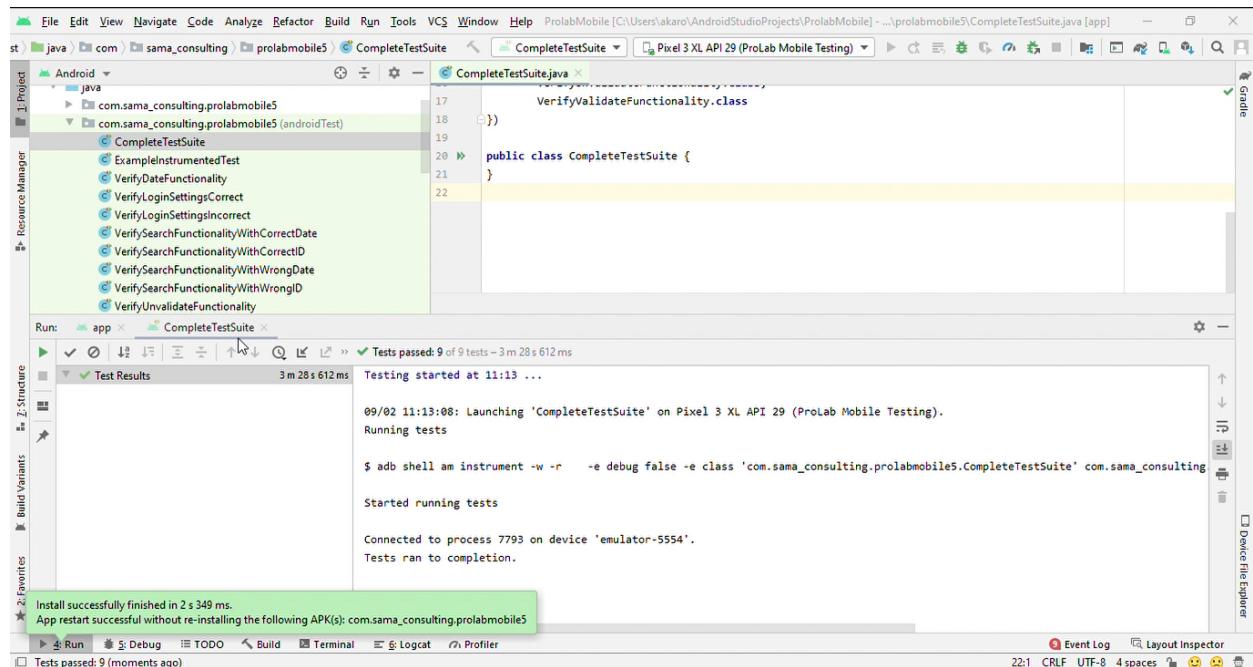
```

Test Results: 26s 615 ms Testing started at 11:13 ...  
 com.sama\_consulting.prolabmobile5 26s 615 ms  
 verify.LoginSettingsCorrect 26s 615 ms  
 08/30 11:13:51: Launching 'VerifyLoginSettingsCorrect' on Pixel 3 XL API 29 (ProLab Mobile Testing).  
 Running tests  
\$ adb shell am instrument -w -r -e debug false -e class 'com.sama\_consulting.prolabmobile5.VerifyLoginSettingsCorrect' com.sama\_consulting.prolabmobile5  
Connected to process 28042 on device 'emulator-5554'.

```

1 package com.sama_consulting.prolabmobile5;
2
3 import org.junit.runner.RunWith;
4 import org.junit.runners.Suite;
5
6 @RunWith(Suite.class)
7 @Suite.SuiteClasses({
8     VerifyLoginSettingsIncorrect.class,
9     VerifyLoginSettingsCorrect.class,
10    VerifyDateFunctionality.class,
11    VerifySearchFunctionalityWithCorrectDate.class,
12    VerifySearchFunctionalityWithCorrectID.class,
13    VerifySearchFunctionalityWithWrongDate.class,
14    VerifySearchFunctionalityWithWrongID.class,
15    VerifyUnvalidateFunctionality.class,
16    VerifyValidateFunctionality.class
17 })
18 public class CompleteTestSuite {
19
20 }
21
22

```



## 3.5 Conclusion

Dans ce troisième chapitre, nous avons installé l'environnement Android Studio et appris à utiliser son Framework de Test Espresso puis nous avons créé une application de Test pour vérifier que notre environnement fonctionne correctement enfin on a traduit les Test Cases qu'on a préparé dans le premier chapitre pour atteindre le but de notre stage de faire un Test Case automatique sur l'application ProLab Mobile 5.

# Conclusion et Perspectives

Notre objectif au cours de stage est d'apprendre de nouveau compétence surtout au niveau sociale et d'apprendre à utiliser Android Studio et son Framework de Test Espresso sur une application nommée ProLab qui gère les fichiers des patient d'une laboratoire médical à travers une application mobile.

Comme pour tout travail de développement qui doit être efficace et qui suit les règles de l'art, nous sommes passées par des différentes étapes pour apprendre et maîtriser l'intégration et le Testing de cette application.

Ce travail nous a permis d'apprendre énormément sur l'environnement Android Studio et sur le Framework Espresso en utilisant la langage Java.

En guise de conclusion, nous avons pris soin de mettre en place les connaissances acquises pendant la période de la formation académique tout en suivant les bonnes pratiques de la programmation en technologie utilisée pendant les années de nos études .

A la fin de notre projet, nous avons pu achever la plupart des missions que nous avons planifiées et spécifiées malgré les contraintes du temps et les difficultés techniques que nous avons rencontrées.

Pour conclure, nous jugerons assez correcte d'estimer que les objectifs du projet ont été réalisés mais il est important de noter que notre travail est loin d'atteindre la perfection et reste ouvert à toute proposition d'amélioration

# Bibliographie

- [1] S. Consulting, “[www.sama-consulting.com](http://www.sama-consulting.com).”
- [2] N. Communications, “The mobile advantage,” 2020.
- [3] ISTQB, “International software testing qualifications board / [www.istqb.org](http://www.istqb.org),” 2002.
- [4] ISTQB, “Certified tester specialist istqb® mobile application testing foundation level syllabus version 2019 / [www.istqb.org/documents/ctfl-mat-syllabus-2019.pdf](http://www.istqb.org/documents/ctfl-mat-syllabus-2019.pdf),” 2019.
- [5] ISTQB, “Certified tester foundation level syllabus version 2018 v3.1 / [www.istqb.org/downloads/send/2-foundation-level-documents/281-istqb-ctfl-syllabus-2018-v3-1.html](http://www.istqb.org/downloads/send/2-foundation-level-documents/281-istqb-ctfl-syllabus-2018-v3-1.html),” 2018.
- [6] A. Studio, “Android espresso / [developer.android.com/training/testing/espresso](http://developer.android.com/training/testing/espresso),” 2020.
- [7] Vogella, “Définition du framework espresso / [vogella.com/tutorials/androidtestingespresso/article.html](http://vogella.com/tutorials/androidtestingespresso/article.html),” 2020.
- [8] experitest, “Comparaison entre espresso et appium / [experitest.com/mobile-app-testing/getting-started-espresso](http://experitest.com/mobile-app-testing/getting-started-espresso),” 2002.