



République Tunisienne  
Ministère de l'enseignement Supérieur, de la recherche scientifique et de la  
technologie  
Direction Général des Etudes Technologiques  
Institut Supérieur des Etudes Technologique de Charguia



# Rapport de Stage de Perfectionnement

**Elaboré par :**

**BOUSSETTA Roua**


**Encadré par :**

**Mme. Chakroun Souhir**

**Société d'accueil : Ysolutions**

**Année universitaire 2017/2018**

# Remerciements



Au terme de ce stage je tiens à exprimer mes respects, mes vifs et mes sincères remerciements pour M. Bouali Hassen qui est intervenu pour être accepté dans la société et pour toutes personnes contribuant de près ou de loin à l'élaboration de cet humble travail et ce modeste rapport.

Je tiens à remercier très vivement M. Chakroun Mohamed Amine qui m'a donné le droit de passer ce stage dans cette société et qui m'a donné plus de confiance et d'encouragement pour faire mieux.

Je m'intéresse aussi à exprimer ma profonde gratitude et mes sincères remerciements à mon encadrant Mme. Masmoudi Souhir pour ses précieux conseils.

# Sommaire

<b>INTRODUCTION GENERALE .....</b>	<b>1</b>
<b>CHAPITRE 1 : PRESENTATION DU CADRE DU STAGE.....</b>	<b>3</b>
<b>Introduction .....</b>	<b>3</b>
<b>1. Présentation de l'entreprise.....</b>	<b>3</b>
1.1 Partie intégration : .....	3
1.1.1 Hardware & Infrastructures .....	3
1.1.2 Solutions de sécurité physique .....	3
1.1.3. Développement de softwares .....	4
1.2 Partie consulting : .....	4
<b>2. Cadre du projet .....</b>	<b>5</b>
2.1 Etude de l'existant .....	5
2.2 Critique de l'existant .....	5
2.3. Solution proposée .....	5
2.3.1. Objectif .....	5
<b>Conclusion.....</b>	<b>5</b>
<b>CHAPITRE 2 : SPECIFICATION DES BESOINS.....</b>	<b>7</b>
<b>Introduction .....</b>	<b>7</b>
<b>1. Besoins fonctionnels .....</b>	<b>7</b>
<b>2. Besoins non fonctionnels .....</b>	<b>7</b>
<b>3. Diagrammes de cas d'utilisation .....</b>	<b>7</b>
3.1 Présentation des acteurs .....	8
3.2 Description des cas d'utilisation .....	8
<b>4. Diagramme de séquence .....</b>	<b>9</b>
<b>5. Conclusion.....</b>	<b>9</b>
<b>CHAPITRE 3 : ETUDE CONCEPTUELLE &amp; REALISATION DU SYSTEME .....</b>	<b>11</b>
<b>Introduction .....</b>	<b>11</b>

<b>1. Aperçu sur le système d’alarme conçu .....</b>	<b>11</b>
<b>2. Schéma synoptique.....</b>	<b>11</b>
<b>3. Environnement de développement du système anti-intrusion .....</b>	<b>12</b>
3.1 Environnements matériels .....	12
3.1.1. La carte Arduino Méga 2560 .....	12
3.1.2. Dispositifs de commande .....	14
3.1.2.1 Clavier.....	14
3.1.2.2. Afficheur LCD .....	15
3.1.2.3. Horloge DS1307 .....	16
3.1.3. Capteur de mouvement PIR : .....	16
3.1.4. Dispositif destiné à donner l'alerte .....	19
3.1.4.1. Module GSM simA6.....	19
3.2. Environnements logiciels .....	19
3.2.1. Android Studio.....	19
3.2.1.1. Le fonctionnement de l’application .....	20
<b>Conclusion.....</b>	<b>21</b>
<b>CONCLUSION GENERALE .....</b>	<b>22</b>
<b>BIBLIOGRAPHIE ET NETOGRAPHIE.....</b>	<b>23</b>

# Liste des figures

<b>Figure 1 : Aperçu sur l'Activité de « Y solutions » .....</b>	<b>4</b>
<b>Figure 2 : Diagramme de cas d'utilisation .....</b>	<b>8</b>
<b>Figure 3 : Diagramme de séquence.....</b>	<b>9</b>
<b>Figure 4 : schéma synoptique du système d'alarme anti-intrusion .....</b>	<b>12</b>
<b>Figure 5 : Dispositif de commande pour les centrales d'alarme .....</b>	<b>14</b>
<b>Figure 6 : Dispositif de commande pour les centrales d'alarme à distance .....</b>	<b>14</b>
<b>Figure 7 : Clavier 16 touches.....</b>	<b>15</b>
<b>Figure 8 : Afficheur LCD 20*4 .....</b>	<b>15</b>
<b>Figure 9 : Horloge temps réel DS1307.....</b>	<b>16</b>
<b>Figure 10 : Détecteur de mouvement PIR.....</b>	<b>17</b>
<b>Figure 11 : Fonctionnement d'un PIR.....</b>	<b>17</b>
<b>Figure 12 : Branchement d'un PIR .....</b>	<b>18</b>
<b>Figure 13 : Réglage d'un PIR.....</b>	<b>18</b>
<b>Figure 14 : Module GSM Sim A6 .....</b>	<b>19</b>
<b>Figure 15: interface d'accueil .....</b>	<b>20</b>
<b>Figure 16 : Description de l'application .....</b>	<b>20</b>
<b>Figure 17 : Interface d'activation et désactivation du système .....</b>	<b>21</b>
<b>Figure 18 : l'envoi du code vers Arduino .....</b>	<b>21</b>

# **Introduction générale**

Dans le cadre de ma scolarité à l'institut Supérieur des études Technologiques de Charguia 2, j'ai eu l'honneur d'effectuer un stage de perfectionnement de quatre semaines du 02/07/2018 jusqu'au 31/07/2018 au sein de la société YSOLUTIONS à Soukra.

Mon stage s'est déroulé au service de sécurité, et grâce à cette expérience pratique, j'ai eu l'opportunité de mettre en pratique mes compétences théoriques.

Je vous expose dans ce rapport en premier lieu une présentation de l'entreprise là où j'ai effectué mon stage.

En deuxième lieu, je vous explique la spécification des besoins fonctionnels et non fonctionnels et les diagrammes des cas d'utilisation et de séquence qui simplifie l'utilisation de système réalisé.

Par suite, je vous présente l'étude conceptuelle et la réalisation du système, et je fini par une conclusion générale là où j'ai annoncé mon objectif du ce stage et une petite récapitulation à propos de mon travail.

***Chapitre 1 :***  
***Présentation du cadre***  
***de stage***

# **Chapitre 1 : Présentation du cadre du stage**

## **Introduction**

La nécessité de protéger son domicile n'est pas nouvelle. Depuis plusieurs siècles déjà, les populations du monde entier ont développé des systèmes électroniques embarqués destinés à prévenir toute intrusion.

L'histoire du système d'alarme a commencé bien avant notre ère et il est toujours en cours.

## **1. Présentation de l'entreprise**

Lancée en 2010, « Y solutions »<sup>1</sup> est devenu, malgré son jeune âge, un acteur majeur en Tunisie dans le domaine du développement et l'intégration IT<sup>2</sup> des réseaux d'entreprises et du consulting. En effet, « Y solutions » propose des solutions qui couvrent l'ensemble des besoins IT des entreprises :

### **1.1 Partie intégration :**

#### **1.1.1 Hardware & Infrastructures**

L'objectif est de fournir des solutions complètes aux entreprises et aux organisations gouvernementales répondant à leur besoin en plateformes et infrastructures. En effet, le développement d'une infrastructure IT solide est aujourd'hui un facteur clé de réussite pour toute entreprise, quel que soit son secteur d'activité.

#### **1.1.2 Solutions de sécurité physique**

Fournir et mettre en place des solutions de sécurité physique. Ces solutions incluent les contrôles d'accès, les systèmes de vidéosurveillance Analogique et IP, les systèmes anti intrusion, les systèmes de détection d'incendie d'extinction automatique et Détection de Gaz, etc.

---

<sup>1</sup> Le site officiel de la société : <http://www.ysolutions.com.tn> .

<sup>2</sup> Abréviation du mot informatique, la technologie de l'information.



### 1.1.3. Développement de softwares

Une équipe qualifiée répond aux besoins de certains clients en effectuant des développements spécifiques permettant de compléter les produits « standards ».

Les domaines d'intervention de l'équipe développement couvrent à titre d'exemple :

- Développement Web
- Développement d'Application Mobile
- Développement de Solutions Logiciels sur Mesure

## 1.2 Partie consulting :

Fort de son expérience à l'échelle nationale, « Y Solutions » compte aujourd'hui une équipe de consultants avec une expertise confirmée principalement sur le marché des Télécommunications et Services.

Globalement, l'approche de « Y solutions » est « orientée marché », tenant compte des clients, des canaux de distribution ainsi que des pratiques les plus compétitives. Le succès de ses recommandations se mesure à leur aptitude à passer le cap d'une implémentation pratique dans le monde réel.



Figure 1 : Aperçu sur l'Activité de « Y solutions »

## **2. Cadre du projet**

### **2.1 Etude de l'existant**

Fort d'une expérience de plus de 8 ans en systèmes d'alarme, Ysolutions développe des solutions de protection éprouvées et intégrant les dernières innovations technologiques.

### **2.2 Critique de l'existant**

Les centrales d'alarme commercialisées, en Tunisie, sont relativement chères.

### **2.3. Solution proposée**

Y solutions cherche à créer une unité de production pour construire sa propre centrale d'une bonne qualité et pas trop chère. Son objectif est de vendre ces centrales et donner des prix compétitifs sur le marché local pour garantir une protection efficace et pas trop chère pour les clients, basique sur les petits budgets (inférieurs aux prix des fournisseurs).

#### **2.3.1. Objectif**

L'objectif de l'entreprise est de produire une centrale d'alarme qui permet de surveiller un local ou un petit appartement avec un budget minimal.

Donc il faut étudier, programmer et réaliser un système d'alarme filaire avec la carte Arduino. Un système qui détecte le mouvement des personnes.

Notre mission est de créer une application Android qui permet de commander à distance le système d'alarme qui a été programmé avec la carte Arduino.

## **Conclusion**

Tout au long de ce chapitre, j'ai présenté l'organisme d'accueil et également j'ai décrit le cadre du travail et détaillé mon objectif. Dans le chapitre suivant, je vais présenter l'étude d'une centrale d'alarme et son principe de fonctionnement.

*Chapitre 2 :*  
*Spécification des*  
*besoins*

# Chapitre 2 : Spécification des besoins

## Introduction

Pendant la période de mon stage j'ai fait des tâches de réalisation et de création d'une application mobile pour commander à distance un système d'alarme programmé par Arduino.

Pour cela je définirai ces tâches dans ce chapitre.

## 1. Besoins fonctionnels

Avant la réalisation du système et la création de l'application, il faut définir quelques besoins fonctionnels

- Partie pour client en local
- Partie pour client à distance
- La sécurité
- La facilité d'utilisation

## 2. Besoins non fonctionnels

Les besoins non fonctionnels décrivent toutes les contraintes auxquelles est soumis le système pour sa réalisation et son bon fonctionnement

- L'autonomie
- La rapidité
- L'efficacité

## 3. Diagrammes de cas d'utilisation

Le diagramme de cas d'utilisation est présenté dans la figure suivante :

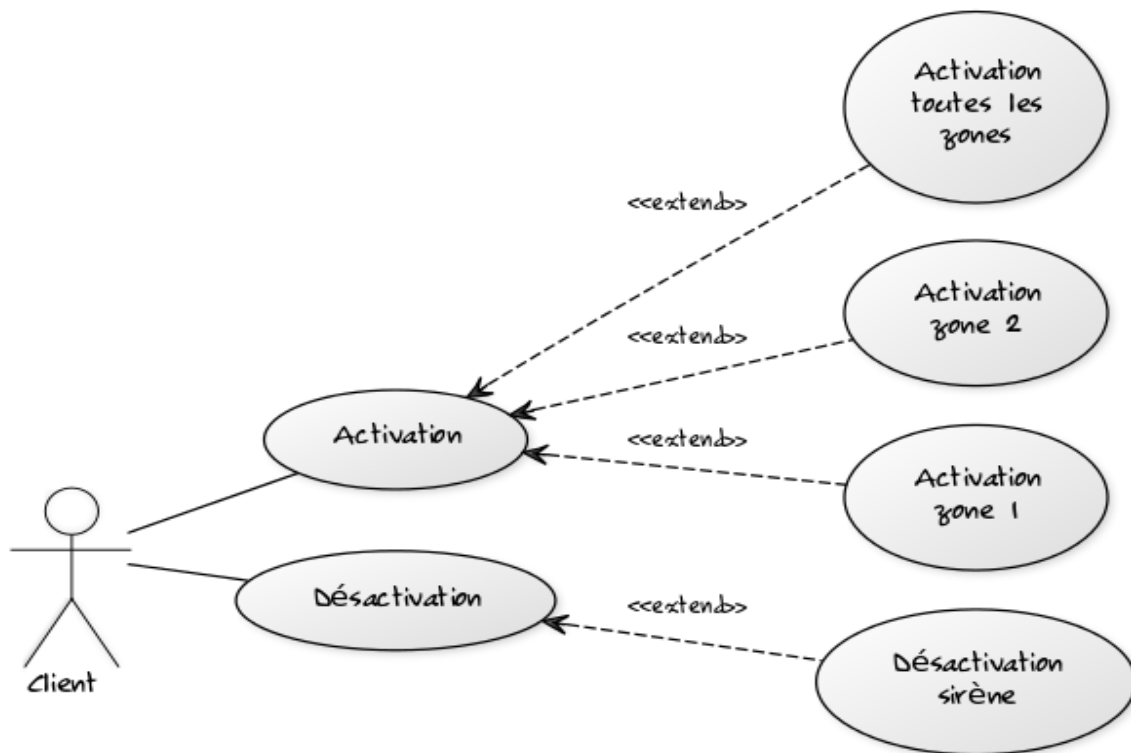


Figure 2 : Diagramme de cas d'utilisation

### 3.1 Présentation des acteurs

Les acteurs du système sont les clients qui peuvent à partir du clavier s'ils sont en local ou bien à partir d'une application mobile s'ils sont loin du système, d'activer ou désactiver une alarme.

### 3.2 Description des cas d'utilisation

Un cas d'utilisation est utilisé pour définir le comportement du système. Chaque cas d'utilisation spécifie une séquence d'action, y compris des variantes, que l'entité réalise, en collaboration avec les acteurs de l'entité.

Dans mon application, les cas d'utilisation sont :

- **Activation des zones** : Avec un simple clic sur un bouton Activer zone 1 ou Activer zone 2 ou bien Activer toutes les zones, le système va répondre au besoin du client.

- **Désactivation des zones :** Avec un simple clic sur un bouton Désactiver zones l'alarme va être désactivé.

## 4. Diagramme de séquence

Le diagramme de séquence est illustré dans la figure ci-dessous :

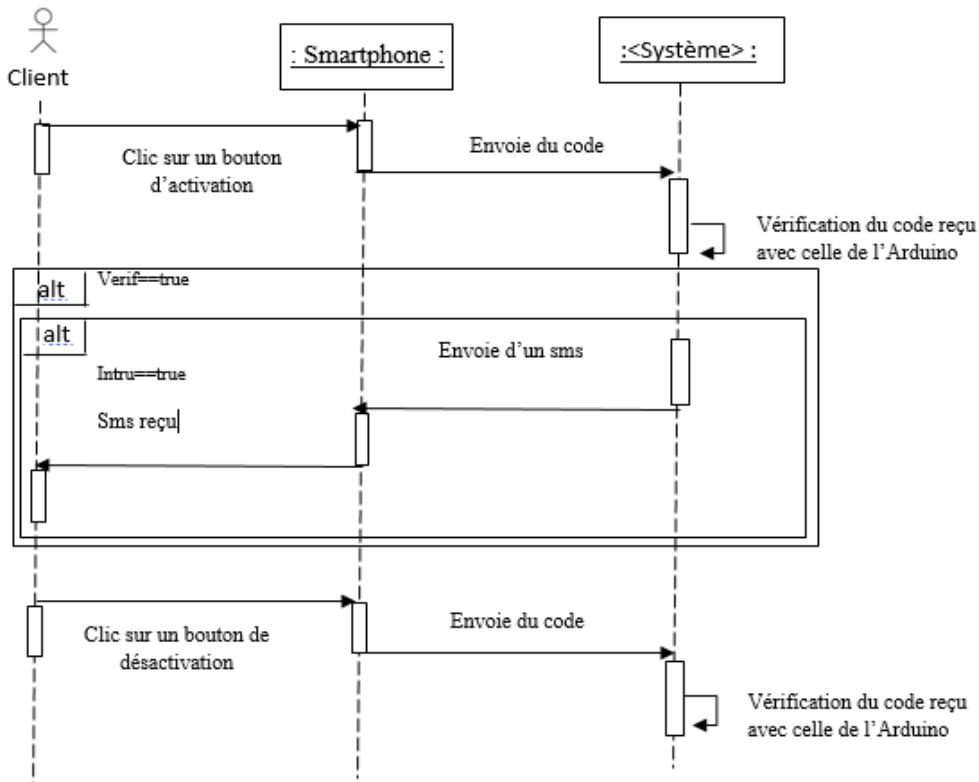


Figure 3 : Diagramme de séquence

## 5. Conclusion

Dans ce chapitre j'ai cité les besoins fonctionnels et non fonctionnels qui sont indispensables pour mieux faciliter le travail à réaliser ainsi que le diagramme de cas d'utilisation, sa description, la présentation des acteurs et j'ai terminé par le diagramme de séquence

# *Chapitre 3 :*

## *Etude conceptuelle & réalisation du système*

# **Chapitre 3 : Etude conceptuelle & réalisation du système**

## **Introduction**

Ce chapitre est consacré à l'étude conceptuelle du système d'alarme, je vais présenter les solutions matérielles de la mise en œuvre de la démarche à suivre dans ce projet.

## **1. Aperçu sur le système d'alarme conçu**

Mon système d'alarme permet de sécuriser un local contre l'intrusion. Afin de le concrétiser, nous allons passer par les étapes suivantes :

- La réalisation d'un système d'alarme filaire et sans fil.
- La création d'une application mobile pour ce système.
- Test du fonctionnement du système avec le clavier.
- Test du fonctionnement du système à distance par l'application mobile.

Les contraintes à respecter sont :

- Le système doit coûter le moins cher possible.
- Le système doit être facile à mettre en œuvre.
- Le système doit être évolutif.
- Le système doit-être simple à utiliser.

## **2. Schéma synoptique**

La figure suivante présente les composants nécessaires pour réaliser le système d'alarme avec Arduino :



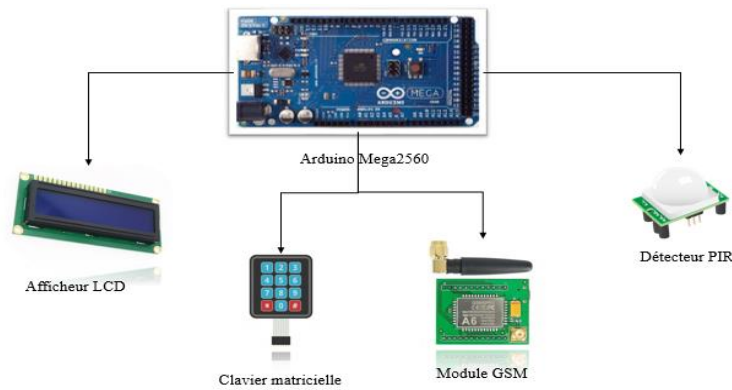


Figure 4 : schéma synoptique du système d'alarme anti-intrusion

### 3. Environnement de développement du système anti-intrusion

Une alarme anti-intrusion est un système conçu pour détecter toute entrée non autorisée dans un bâtiment ou une zone.

#### 3.1 Environnements matériels

##### 3.1.1. La carte Arduino Méga 2560

La carte Arduino<sup>3</sup>, dans notre système, constitue la centrale d'alarme qui centralise les informations envoyées par les détecteurs et prend la décision de lancer l'alerte.

##### Pourquoi Arduino ?

Il y a de nombreuses cartes électroniques qui possèdent des plateformes basées sur des microcontrôleurs disponibles pour l'électronique programmée. Tous ces outils prennent en charge les détails compliqués de la programmation et les intègrent dans une présentation facile à utiliser. De la même façon, le système Arduino simplifie la façon de travailler avec les microcontrôleurs tout en offrant aux personnes intéressées plusieurs avantages cités comme suit:

➤ **Le prix (réduits):** les cartes Arduino sont relativement peu coûteuses comparativement

---

<sup>3</sup> <https://www.gotronic.fr/art-carte-arduino-mega-2560-12421.htm>  
[https://zestedesavoir.com/tutoriels/686/arduino-premiers-pas-en-informatique-embarquee/742\\_decouverte-delarduino/3414\\_presentation-darduino/](https://zestedesavoir.com/tutoriels/686/arduino-premiers-pas-en-informatique-embarquee/742_decouverte-delarduino/3414_presentation-darduino/)

aux autres plates-formes.

➤ **Multi plateforme** : le logiciel Arduino, écrit en JAVA, tourne sous les systèmes d'exploitation Windows, Macintosh et Linux. La plupart des systèmes à microcontrôleurs sont limités à Windows.

➤ Un environnement de programmation clair et simple: l'environnement de programmation Arduino (le logiciel Arduino IDE) est facile à utiliser pour les débutants, tout en étant assez flexible pour que les utilisateurs avancés puissent en tirer profit également.

➤ **Logiciel Open Source et extensible** : le logiciel Arduino et le langage Arduino sont publiés sous licence open source, disponible pour être complété par des programmeurs expérimentés.

➤ **Matériel Open source et extensible** : les cartes Arduino sont basées sur les Microcontrôleurs ATmel ATMEGA8, ATMEGA168, ATMEGA 328, les schémas des modules sont publiés sous une licence créative Commons, et les concepteurs des circuits expérimentés peuvent réaliser leur propre version des cartes Arduino, en les complétant et en les améliorant. Même les utilisateurs relativement inexpérimentés peuvent fabriquer la version sur plaque d'essai de la carte Arduino, dont le but est de comprendre comment elle fonctionne pour économiser le coût.

## **Les caractéristiques**

La carte Arduino Méga possède les Caractéristiques suivantes :

- Microcontrôleur : ATMEGA2560
- Tension de fonctionnement : 5v
- Tension d'alimentation (recommandée) 7-12v.
- Tension d'alimentation (limites) 6-20v.
- Nombre Entrées/Sorties numériques : 54(dont 15 fournissent des sortie PWM)
- Nombre de port analogique : 16
- Courant max par E/S : 50mA
- Courant pour broches : 3.3v 80mA
- Mémoire Flash : 256Ko
- EEPROM : 4Kb
- Vitesse horloge : 16MHZ

Il permet aux utilisateurs de mettre le système en marche ou en arrêt. Il est composé d'un clavier et un afficheur LCD en local et d'une application mobile à distance.



Figure 7 : Clavier 16 touches

### 3.1.2.2. Afficheur LCD

J'ai utilisé un afficheur LCD 20\*4<sup>4</sup> pour afficher des informations utiles pour le client : heure, date d'intrusion et zone.

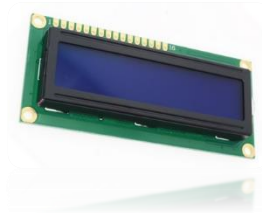


Figure 8 : Afficheur LCD 20\*4

Tableau 1 : Les broches d'un afficheur LCD 20\*4

Numéro du broche	Symbole	Description
1	VSS	GND
2	VDD	Alimentation positive 5V
3	V0	Réglage du contraste entre 0V et 5V
4	RS	Signal de sélection de données / instructions
5	R/W	Lecture / écriture du signal de sélection
6	E	Activer le signal
7-14	DB0-DB7	Ligne de bus de données bidirectionnels
15	A	Anode (5V)
16	K	Cathode (masse)

<sup>4</sup> . <https://www.aurel32.net/elec/lcd.php>

### 3.1.2.3. Horloge DS1307

La fonction horloge / calendrier fournit les informations secondes, minutes, heures, jour, date, mois, et l'année. La fin de la date de mois est ajustée automatiquement pour les mois de moins de 31 jours, y compris des corrections pour l'année bissextile. L'horloge fonctionne soit dans le format 24 heures ou 12 heures avec indicateur AM / PM.

Le DS1307<sup>5</sup> est une horloge RTC<sup>6</sup> série d'une faible puissance, on retrouve une information binaire, codé décimal (BCD) de l'horloge et du calendrier ainsi que 56 octets de secours NV SRAM<sup>7</sup>.

L'adresse et les données sont transférées en série par un bus bidirectionnel I2C.



Figure 9 : Horloge temps réel DS1307

#### Les caractéristiques

- Alimentation : 4,5V à 5,5V
- Consommation : 1,5 mA
- Interface : I2C

### 3.1.3. Capteur de mouvement PIR :

PIR<sup>8</sup> permet de détecter le mouvement d'un corps humain (en effet, la chaleur du corps produit suffisamment de lumière infrarouge pour être mesurée). Le capteur peut détecter un mouvement jusqu'à une distance maximale de 6m.

---

<sup>5</sup> <http://tiptopboards.com/88-horloge-temps-r  el-pour-arduino-ds1307.html>

<sup>6</sup> Horloge temps r  el.

<sup>7</sup> Nv SRAM est un type de m  moire vive non volatile, son fonctionnement est similaire    celui de la m  moire vive statique (SRAM).

<sup>8</sup> Capteur passif d'infrarouge

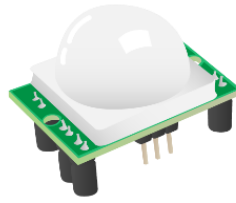


Figure 10 : Détecteur de mouvement PIR

## Principe de fonctionnement du capteur de mouvement PIR

Son rôle est de détecter des présences anormales dans un environnement. Ainsi, il perçoit les formes, les déplacements ou les volumes en utilisant la technique de l'infrarouge. Cet appareil a un rôle sécuritaire, et doit prémunir contre d'éventuels vols ou agressions. Le choix de son emplacement est vital. Son principe de fonctionnement<sup>9</sup> est d'être sensible à la chaleur dégagée par un être vivant. Le détecteur de mouvement transmet toutes les informations enregistrées en temps réel par un système radio permettant l'intervention rapide de la société de surveillance.

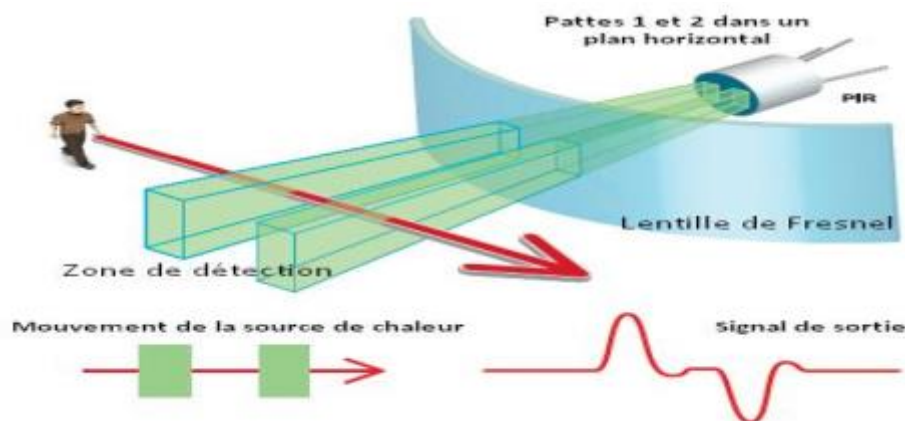


Figure 11 : Fonctionnement d'un PIR

<sup>9</sup> [http://technomoussi.free.fr/IMG/pdf/TP-D1\\_Detecteur\\_de\\_mouvement.pdf](http://technomoussi.free.fr/IMG/pdf/TP-D1_Detecteur_de_mouvement.pdf)

## Branchement

On branche donc :

- Le VCC du PIR sur le 5V de l'Arduino
- Le GRD du PIR sur le GRD de l'Arduino
- La dernière branche sur le pin l'Arduino



Figure 12 : Branchement d'un PIR

## Réglages du PIR

**SX** = Ajustement de la Sensibilité du capteur de 3-7m, visser pour augmenter

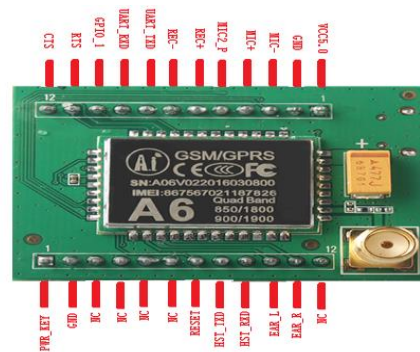
**TX** = Ajustement du délai (Time) pendant lequel la sortie reste verrouillée sur HIGH après une détection de mouvement (TX). Visser pour augmenter la durée, jusqu'à 200 secondes.



Figure 13 : Réglage d'un PIR

#### 3.1.4.1. Module GSM simA6

#### 3.1.4.1. Module GSM simA6



**Figure 14 : Module GSM Sim A6**

## Branchement

- Connecter le pin « power key » avec le VCC
- Connecter RX de module avec le TX d'Arduino
- Connecter TX de sim A6 avec le RX d'Arduino

### 3.2. Environnements logiciels

### 3.2.1. Android Studio

Android Studio est un environnement de développement pour faire la création et le développement des applications Android.

<sup>10</sup> [https://www.elecrow.com/wiki/index.php?title=A6\\_GPRS/GSM\\_Shield](https://www.elecrow.com/wiki/index.php?title=A6_GPRS/GSM_Shield)



### 3.2.1.1. Le fonctionnement de l'application

La première interface est dite la page d'accueil de notre application



Figure 15: interface d'accueil


Le clic sur le bouton  une page de la description de l'application s'affiche.



Figure 16 : Description de l'application

Le clic sur le bouton  amène à la page d'accueil.


Le clic sur le bouton  permet d'afficher la page où on trouve les boutons d'activation et désactivation du système d'alarme.



Figure 17 : Interface d'activation et désactivation du système

Le clic sur une de ces 5 boutons assure un envoi du code de la zone voulue pour l'activer ou la désactiver après une vérification précise de ce code envoyé avec celle qui est programmé par Arduino.

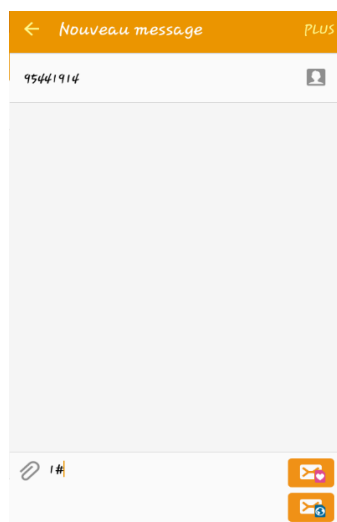


Figure 18 : l'envoi du code vers Arduino

## Conclusion

Dans ce chapitre, j'ai décrit la solution matérielle et logicielle proposée pour la réalisation du système d'alarme ainsi que la présentation des interfaces réalisés dans mon application mobile pour clarifier les étapes d'utilisation de l'application.

# Conclusion générale

Ce travail a été réalisé dans le cadre de mon stage de perfectionnement d'un mois au sein de la société « Ysolutions ».

Ce stage était une bonne occasion pour élargir mes connaissances pratiques pour s'adapter aux nouvelles technologies qui s'améliore jour après jour pour persister dans un secteur qui s'éveille chaque jour sur des nouvelles innovations.

Le projet m'a également donné l'occasion de mieux connaître la société et son **service de sécurité informatique**.

Ce stage m'a permis aussi d'étudier un système d'alarme filaire avec la carte Arduino et de créer une application Android pour pouvoir le commander à distance.

# **Bibliographie et Nétographie**

<http://www.ysolutions.com.tn>

<https://www.aurel32.net/elec/lcd.php>

<http://tiptopboards.com/88-horloge-temps-réel-pour-arduino-ds1307.html>

[http://technomoussi.free.fr/IMG/pdf/TP-D1\\_Detecteur\\_de\\_mouvement.pdf](http://technomoussi.free.fr/IMG/pdf/TP-D1_Detecteur_de_mouvement.pdf)

[https://zestedesavoir.com/tutoriels/686/arduino-premiers-pas-en-informatique-embarquee/742\\_decouverte-de-larduino/3414\\_presentation-darduino/](https://zestedesavoir.com/tutoriels/686/arduino-premiers-pas-en-informatique-embarquee/742_decouverte-de-larduino/3414_presentation-darduino/)

# Annexe1 : Le code source de l'application

## Android

### 1. La première interface

#### 1.1. XML :

```
<?xml version="1.0" encoding="utf-8" ?>
<LinearLayout
xmlns:tools="http://schemas.android.com/tools"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    tools:context=".MainActivity"
    android:background="@drawable/p"
    android:orientation="vertical"
    >
    <Button
        android:id="@+id/about"
        android:layout_width="27dp"
        android:layout_height="28dp"
        android:layout_gravity="right"
        android:background="@drawable/about" />

    <TextView
        android:id="@+id/textView"
        android:layout_marginTop="90dp"
        android:gravity="center"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:fontFamily="serif"

        android:textColor="@color/orange"
        android:textSize="60dp"
        android:text="Bienvenu!" />

    <Button
        android:id="@+id/Create"
        android:layout_width="100dp"
        android:layout_height="100dp"
        android:layout_gravity="center"
        android:layout_marginLeft="3dp"
        android:background="@drawable/btn" />

</LinearLayout>
```



## 1.2. Java :

```
public class MainActivity extends AppCompatActivity {

    Button Create, about;
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main);

        Create=(Button) findViewById(R.id.Create);
        about=(Button) findViewById(R.id.about);

        about.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View view) {
                Activity3(view);
            }
        });

        Create.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View view) {
                Activity2(view);
            }
        });
    }

    public void Activity2(View view){

        Intent Main2Activity = new Intent(MainActivity.this,
Main2Activity.class);
        startActivity(Main2Activity);
    }

    public void Activity3(View view){

        Intent Main3Activity = new Intent(MainActivity.this,
Main3Activity.class);
        startActivity(Main3Activity);
    }
}
```

## 2. Interface des boutons

### 2.1. XML

```

<LinearLayout
    xmlns:tools="http://schemas.android.com/tools"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:orientation="vertical"
    android:padding="10dp"
    android:layout_weight="4"
    android:background="@drawable/preview15"
    tools:context=".Main2Activity">

    <LinearLayout
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:layout_weight="1">

        <Button
            android:id="@id/ActiverZone1"
            android:layout_width="288dp"
            android:layout_height="60dp"
            android:text="ActiverZone1"
            android:textStyle="bold"
            android:fontFamily="casual"
            android:layout_marginLeft="30dp"
            android:background="@drawable/b1"

            />

        </LinearLayout>

    <LinearLayout
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:layout_weight="1">

        <Button
            android:layout_width="288dp"
            android:layout_height="60dp"
            android:text="ActiverZone2"
            android:textStyle="bold"
            android:fontFamily="casual"
            android:layout_marginLeft="30dp"
            android:id="@+id/ActiverZone2"
            android:background="@drawable/b2"

            />

        </LinearLayout>

    <LinearLayout
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:layout_weight="1">

        <Button
            android:layout_width="288dp"
            android:layout_height="60dp"
            android:layout_marginLeft="30dp"

```



```
        android:text="Activer toutes Zones"
        android:textStyle="bold"
        android:fontFamily="casual"
        android:background="@drawable/b5"
        android:id="@+id/Activer tt Zones"
    />
</LinearLayout>

<LinearLayout
    android:layout_width="match_parent"
    android:layout_height="wrap_content"
    android:layout_weight="1">

    <Button
        android:layout_width="288dp"
        android:layout_height="60dp"
        android:layout_marginLeft="30dp"
        android:text="Desactiver Zones"
        android:textStyle="bold"
        android:fontFamily="casual"
        android:background="@drawable/b3"

        android:id="@+id/Desactiver Zones"
    />

</LinearLayout>

<Button
    android:layout_width="35dp"
    android:layout_height="35dp"
    android:id="@+id/retour"
    android:background="@drawable/retour"/>

</LinearLayout>
```

## 2.2. Java

```
public class Main2Activity extends AppCompatActivity {

    Button ActiverZone1, ActiverZone2, DesactiverZones, Activer tt Zones, retour;

    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main2);

        ActiverZone1=(Button) findViewById(R.id.ActiverZone1);
        ActiverZone2=(Button) findViewById(R.id.ActiverZone2);
        DesactiverZones=(Button) findViewById(R.id.DesactiverZones);
        Activer tt Zones=(Button) findViewById(R.id.Activer tt Zones);
        retour=(Button) findViewById(R.id.retour);
        retour.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View view) {
                Retour(view);
            }
        });

        ActiverZone1.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View view) {
```



```
        Activer1();
    }
});

    ActiverZone2.setOnClickListener(new View.OnClickListener() {
        @Override
        public void onClick(View view) {
            Activer2();
        }
    });

    ActiverZone3.setOnClickListener(new View.OnClickListener() {
        @Override
        public void onClick(View view) {
            ActiverZones();
        }
    });

    DesactiverZones.setOnClickListener(new View.OnClickListener() {
        @Override
        public void onClick(View view) {
            Desactiver();
        }
    });
}

public void Activer1(){

    Intent sms = new Intent(Intent.ACTION_SENDTO,
Uri.parse("smsto:95441914"));
    sms.putExtra("sms_body", "1");
    startActivity(sms);
}

public void Activer2(){

    Intent sms = new Intent(Intent.ACTION_SENDTO,
Uri.parse("smsto:95441914"));
    sms.putExtra("sms_body", "2");
    startActivity(sms);
}

public void ActiverZones(){

    Intent sms = new Intent(Intent.ACTION_SENDTO,
Uri.parse("smsto:95441914"));
    sms.putExtra("sms_body", "3");
    startActivity(sms);
}
public void Desactiver(){
```

```

        Intent sms = new Intent(Intent.ACTION_SENDTO,
Uri.parse("sms:95441914"));
        sms.putExtra("sms_body", "3");
        startActivity(sms);
    }

    public void Retour(View view){

        Intent MainActivity = new Intent(Main2Activity.this,
MainActivity.class);
        startActivity(MainActivity);
    }
}

```

### 3. Interface de description de l'application

#### 3.1. XML

```

<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:app="http://schemas.android.com/apk/res-auto"
    xmlns:tools="http://schemas.android.com/tools"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:orientation="vertical"
    tools:context=".Main3Activity"
    android:layout_weight="3">
    <Button
        android:layout_width="30dp"
        android:layout_height="25dp"
        android:id="@+id/acceuil"
        android:background="@drawable/acceuil" />

    <LinearLayout
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:layout_weight="1">

        <TextView
            android:layout_width="match_parent"
            android:layout_height="60dp"
            android:fontFamily="sans-serif-condensed"
            android:gravity="center"
            android:text="A propos"
            android:textColor="@color/vert"
            android:textSize="30dp" />

    </LinearLayout>

<LinearLayout
    android:layout_width="match_parent"
    android:layout_height="wrap_content"
    android:layout_weight="1">

```



```

<TextView
    android:layout_width="match_parent"
    android:layout_height="wrap_content"
    android:layout_marginLeft="2dp"
    android:layout_weight="1"
    android:text="Une alarme anti-intrusion est un système conçu pour
détecter toute entrée non autorisée dans un bâtiment ou une zone.
Notre application sert à activer et désactiver l'alarme des zones.
Une simple clic sur un bouton voulu un message va etre envoyer d'une
manière automatique.
"/>
</LinearLayout>

<ImageSwitcher
    android:id="@+id/imageSwitcher"
    android:layout_width="208dp"
    android:layout_height="275dp"
    android:layout_gravity="center" />

<LinearLayout
    android:layout_width="match_parent"
    android:layout_height="wrap_content"
    android:layout_gravity="center"
    android:layout_weight="1">

    <LinearLayout
        android:layout_width="match_parent"
        android:layout_height="66dp"
        android:layout_weight="1">

        <Button
            android:id="@+id/precedent"
            android:layout_width="61dp"
            android:layout_height="58dp"
            android:layout_marginLeft="100dp"
            android:background="@drawable/precedent" />

        <Button
            android:id="@+id/suivant"
            android:layout_width="61dp"
            android:layout_height="58dp"
            android:layout_marginLeft="50dp"
            android:background="@drawable/next" />

    </LinearLayout>

</LinearLayout>

</LinearLayout>

```

### 3.2. Java

```

public class Main3Activity extends AppCompatActivity {
    private ImageSwitcher imageSwitcher;
    private Button buttonPrevious;
    private Button buttonNext;
    private Button acceuil;
    private final String[] imageNames={"a1", "a2", "a3"};

```

```
private int currentIndex;

@Override
protected void onCreate(Bundle savedInstanceState) {
    super.onCreate(savedInstanceState);

    setContentView(R.layout.activity_main3);

    buttonPrevious = (Button) findViewById(R.id.precedent);
    buttonNext = (Button) findViewById(R.id.suivant);
    accueil=(Button) findViewById(R.id.acceuil);
    imageSwitcher = (ImageSwitcher) findViewById(R.id.imageSwitcher);

    // Animation when switching to another image.
    Animation out= AnimationUtils.loadAnimation(this,
android.R.anim.fade_out);
    Animation in= AnimationUtils.loadAnimation(this,
android.R.anim.fade_in);

    // Set animation when switching images.
    imageSwitcher.setInAnimation(in);
    imageSwitcher.setOutAnimation(out);

    //
    imageSwitcher.setFactory(new ViewSwitcher.ViewFactory() {

        // Returns the view to show Image
        // (Usually should use ImageView)
        @Override
        public View makeView() {
            ImageView imageView = new
ImageView(getApplicationContext());

            imageView.setBackgroundColor(Color.LTGRAY);
            imageView.setScaleType(ImageView.ScaleType.CENTER);
            return imageView;
        }
    });

    this.currentIndex=0;
    this.showImage(this.currentIndex);

    buttonPrevious.setOnClickListener(new View.OnClickListener() {
        @Override
        public void onClick(View v) {
            previousImage();
        }
    });

    buttonNext.setOnClickListener(new View.OnClickListener() {
        @Override
        public void onClick(View v) {
            nextImage();
        }
    });

    accueil.setOnClickListener(new View.OnClickListener() {
        @Override
        public void onClick(View view) {
            Acceuil(view);
        }
    })
}
```

```
        });  
  
    }  
  
    private void Accueil(View view){  
  
        Intent MainActivity = new Intent(Main3Activity.this,  
MainActivity.class);  
        startActivity(MainActivity);  
  
    }  
  
    private void previousImage() {  
        if(currentIndex > 0) {  
            currentIndex--;  
        }else {  
            Toast.makeText(getApplicationContext(), "No Previous Image",  
Toast.LENGTH_SHORT).show();  
            return;  
        }  
        this.showImage(currentIndex);  
    }  
  
    private void nextImage() {  
        if(currentIndex < this.imageNames.length-1) {  
            currentIndex++;  
        }else {  
            Toast.makeText(getApplicationContext(), "No Next Image",  
Toast.LENGTH_SHORT).show();  
            return;  
        }  
        this.showImage(currentIndex);  
    }  
  
    private void showImage(int imgIndex) {  
        String imageName= this.imageNames[imgIndex];  
  
        int resId= getDrawableResIdByName(imageName);  
        if(resId!= 0) {  
            this.imageSwitcher.setImageResource(resId);  
        }  
    }  
  
    // Find Image ID corresponding to the name of the image (in the  
drawable folder).  
    public int getDrawableResIdByName(String resName) {  
        String pkgName = this.getPackageName();  
        // Return 0 if not found.  
        int resID = this.getResources().getIdentifier(resName , "drawable",  
pkgName);  
        Log.i("MyLog", "Res Name: " + resName + "==> Res ID = " + resID);  
        return resID;  
    }  
  
}
```