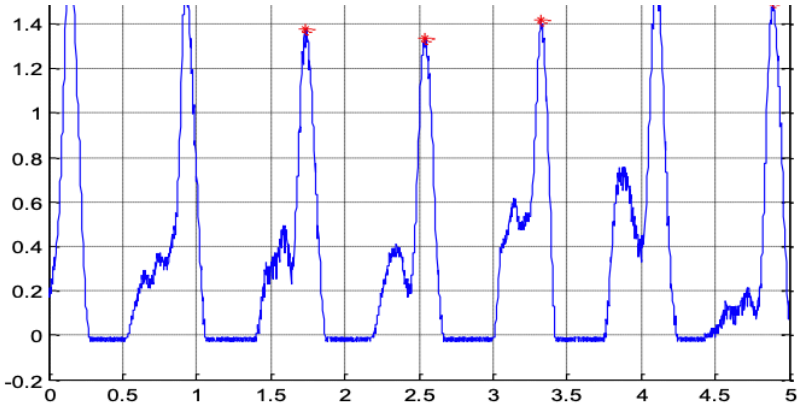
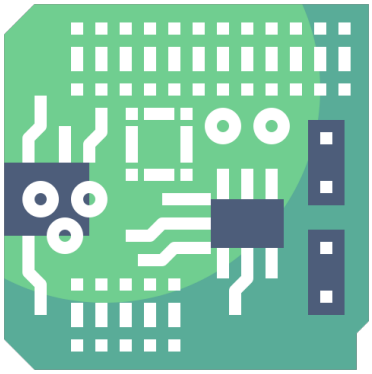
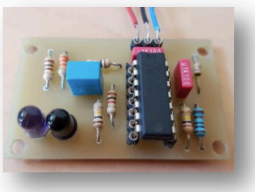
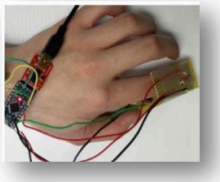


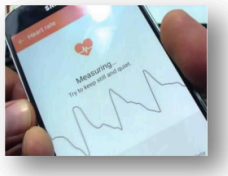




PCB



Acquisition sur le doigt

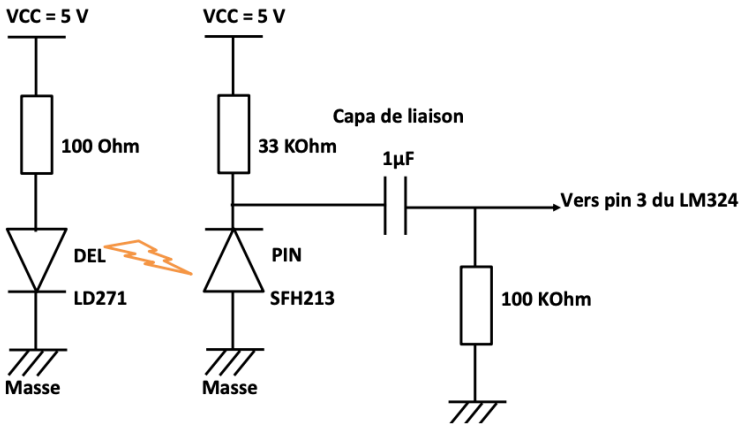


Visualisation sur Smartphone

Ateliers - AT15
Oxymétrie

Alban FERRACANI, Awais CHAUDHRY, Marie ESTIVALS, Inès DJERIDI
E1- 2020/2021

ESIEE PARIS



ESIEE PARIS

Founding Member
Université Gustave Eiffel

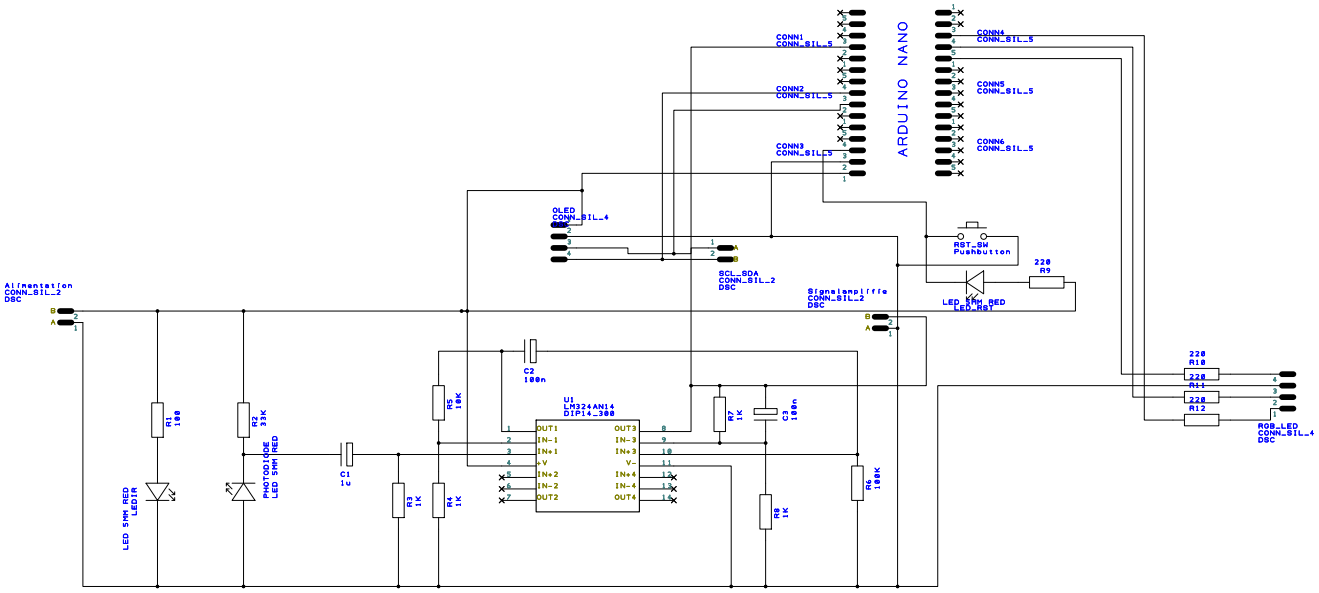
Affiliated to
CCI PARIS ILE-DE-FRANCE EDUCATION

ESIEE PARIS	**PERS – Conception du PCB, miniaturisation, application mobile et boitier 3D.**	08/06/2021 09/06/2021 10/06/2021 12h personnel

Schématisation électrique

Afin de pouvoir concevoir le layout du PCB, nous avons utilisé le logiciel de CAO préconisé par l'ESIEE à savoir DESIGN SPARK.
La première étape consiste à créer le schéma électrique. On choisit d'abord les composants, puis si nécessaire, on saisit leur valeur. Cette saisie n'est pas obligatoire mais nous l'avons faite rigoureusement car elle sera très utile par la suite pour se repérer dans la nomenclature et dans la soudure des composants sur le PCB par la suite.

Le schéma ci-dessous est celui que nous avons effectué sous DESIGN SPARK en vue de la modélisation du PCB. On y repère les composants principaux du montage vu en classe (LM324, diode IR, photodiode, filtrage) mais également quelques composants supplémentaires comme l'écran OLED, un bouton poussoir permettant de remettre à zéro la mesure, un emplacement pour une LED RGB (red-green-blue) ainsi qu'un bornier sur lequel viendra se clipser l'Arduino nano.



Une dernière fonctionnalité que nous avons ajoutée (qui n'apparait pas sur le schéma ci-dessus mais que l'on a tracé directement sur le PCB) est l'ajout d'un régulateur de tension LM7805 afin de pouvoir alimenter le circuit avec une plage de tension comprise entre 5 et 18 Volts. On a également couplé ce régulateur de tension avec un condensateur de filtrage qui permettra d'éliminer les parasites et garder exclusivement un courant continu après filtrage.

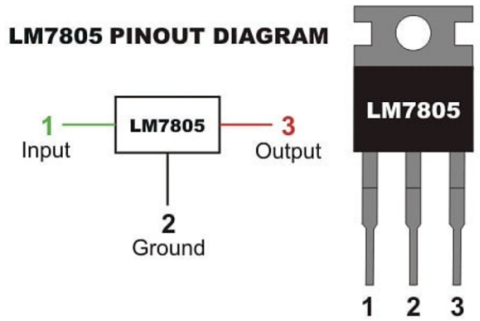
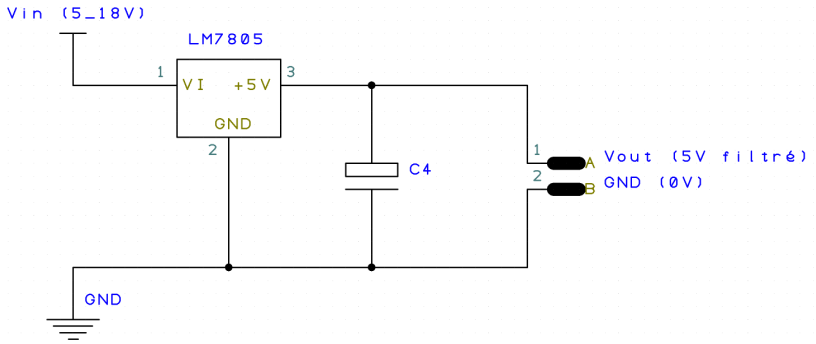
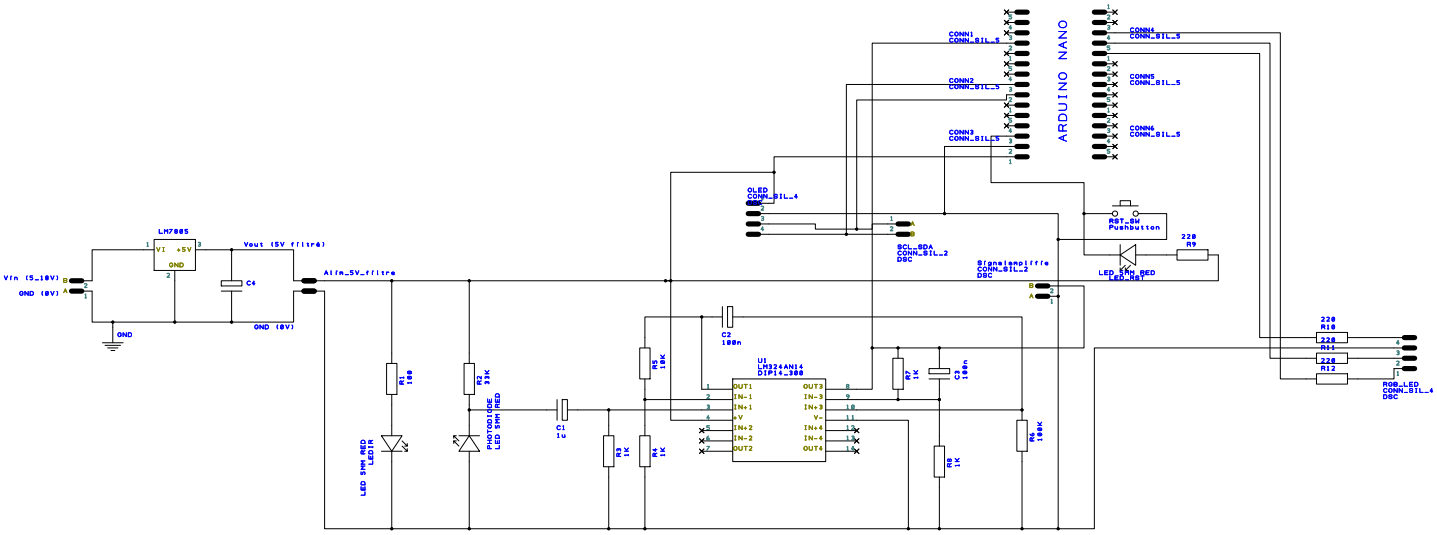


Figure 12 : Datasheet LM7805

Cette technique de filtrage est d'ailleurs largement utilisée à la sortie des ponts de diodes par exemple (pont redresseur) afin de lisser le signal qui n'est pas parfaitement un courant continu. Ci-dessous, le schéma électrique (effectué sous DESIGN SPARK) de la partie alimentation et filtrage de l'alimentation.



Si l'on ajoute maintenant cette partie d'alimentation et de filtrage de l'alimentation au schéma précédent, cela donne le schéma global suivant qui va nous permettre de dessiner le PCB.



Conception du PCB

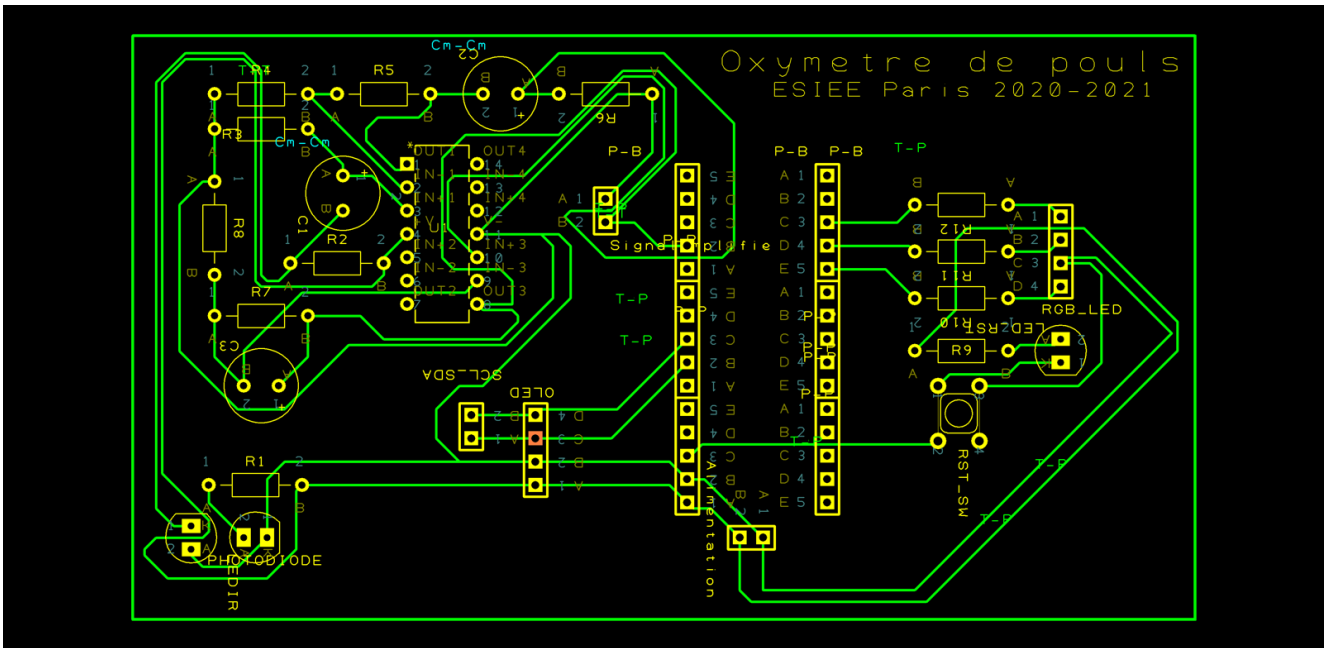
Nous sommes donc aptes à convertir ce schéma en un PCB. Plusieurs contraintes apparaissent ici :

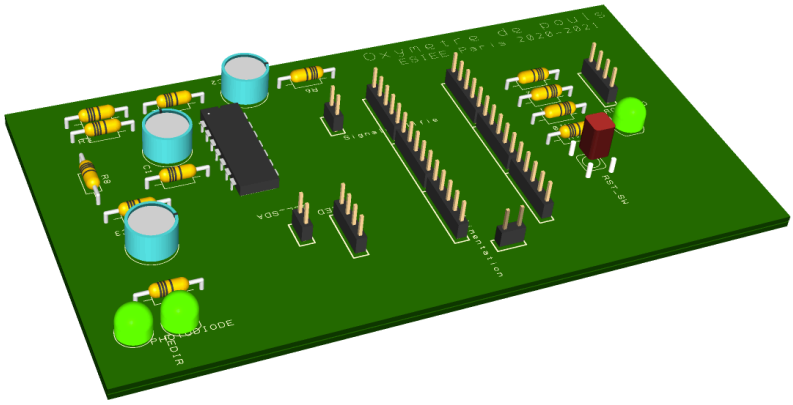
- ⇨ La notion de taille du PCB : celui-ci ne doit pas être trop grand mais doit pouvoir accueillir tous les composants et les pistes nécessaires au bon fonctionnement de l'oxymètre (y compris les composants additionnels comme l'écran OLED, la LED RGB, le bouton poussoir et sa LED ainsi que l'arduino nano et le bloc alimentation).

- ⇨ La notion de pistes qui doivent être en mono couche : le tracé est donc beaucoup plus difficile, d'autant plus que nous avons ajouté beaucoup de composants additionnels. Les pistes ne doivent donc pas se croiser, se chevaucher.

- ⇨ La notion de disposition des composants : les composants doivent être positionnés ergonomiquement. En effet, l'écran doit être dans le même sens que le capteur (diode IR + photodiode) et suffisamment espacé du LM324 (Amplificateur Opérationnel) ainsi que de l'Arduino Nano qui sont plus épais que ce dernier. Enfin, il faut penser à l'intégration de ce PCB dans un boitier imprimé en 3D.

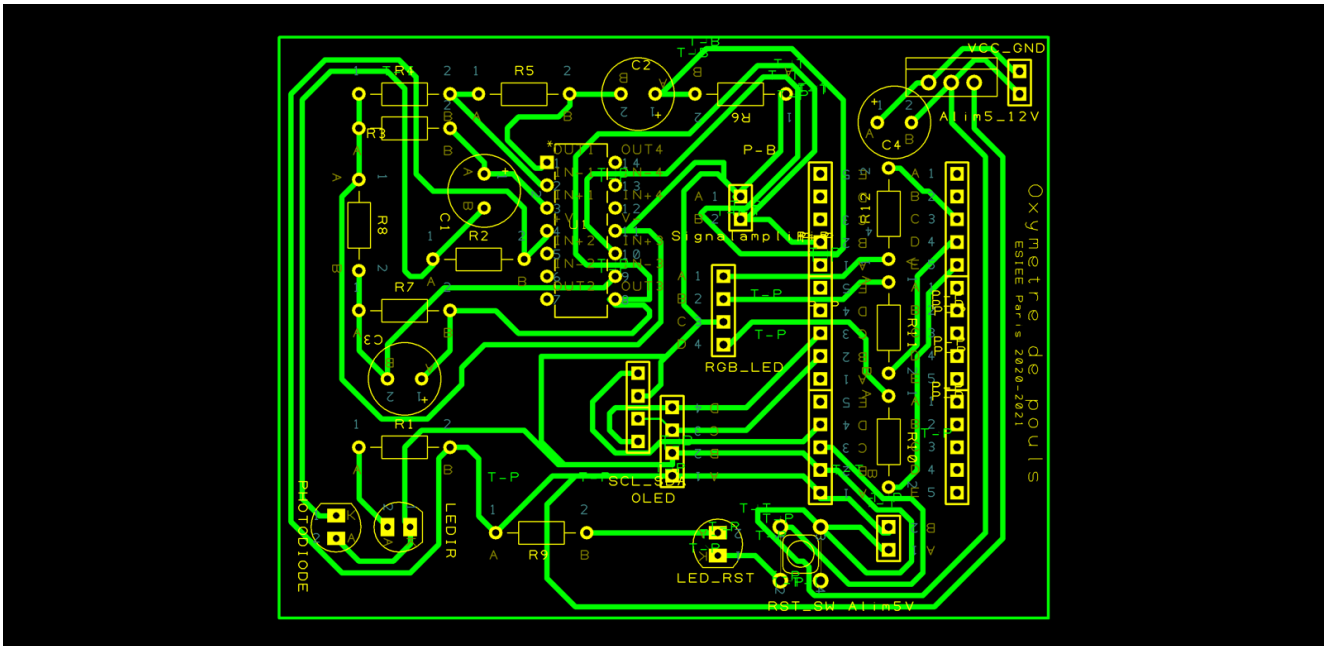
En prenant en compte toutes ces contraintes, nous sommes arrivés à une première version du PCB. En imprimant ce PCB sur papier (à l'échelle), nous avons trouvé que ses dimensions n'étaient pas appropriées et même démesurées. Voici l'illustration de ce premier PCB conçu sur DESIGN SPARK.



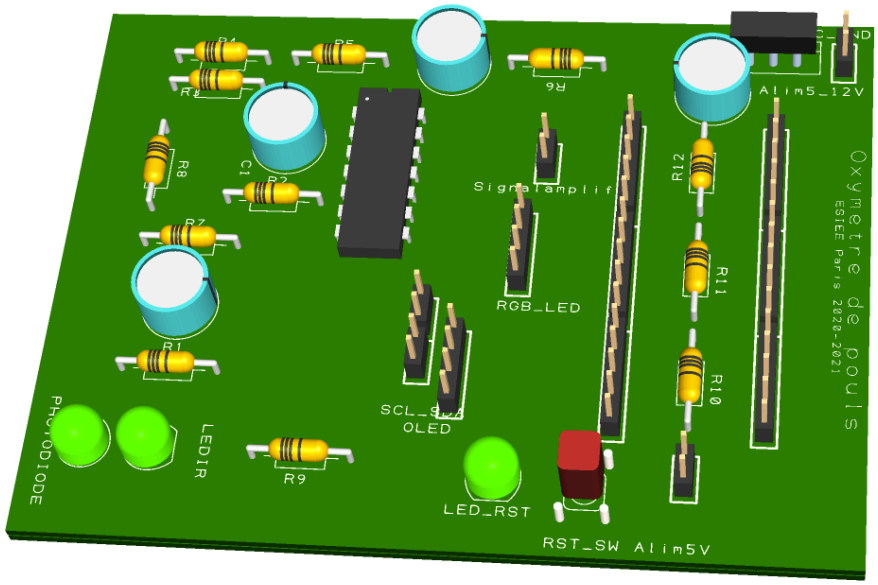


Pour ce qui est de sa modélisation 3D (cf. figure ci-contre), une fois celle-ci imprimée à l'échelle, on se rend encore mieux compte de la disposition des composants (car nous ne faisons pas apparaître les pistes dessus), ce qui nous nous permet de nous focaliser sur l'ergonomie, la disposition des composants et la taille du PCB.

Nous avons donc créé une deuxième version de ce PCB puis une troisième, laquelle est beaucoup plus compacte, l'ergonomie a été étudiée pour même pouvoir positionner l'écran OLED dans différentes orientations. En voici l'aboutissement en termes de schématisation du PCB et de modélisation 3D.



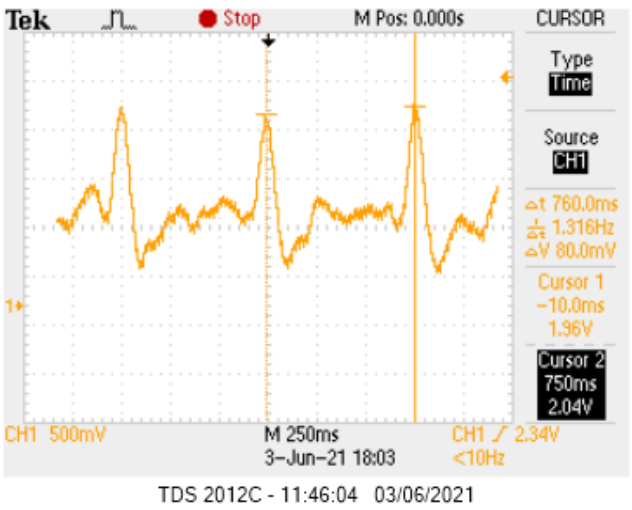
On se rend bien compte qu'on a gagné plus de 30% d'espace environ en optimisant la disposition des composants. L'arduino sera positionné au niveau des deux rangées de pins à droite.

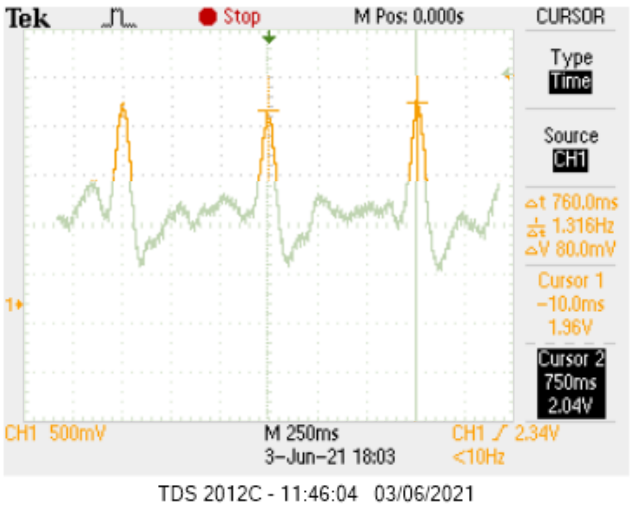


Programme informatique pour l'analyse des signaux

L'objectif premier de l'Arduino est de récupérer le signal analogique amplifié (qui est ici relié sur son pin A0) et de pouvoir le tracer directement sur le moniteur série ou alors de l'exporter dans d'autres applications comme Matlab ou Mit App Inventor par exemple.
Néanmoins, nous avons voulu compléter ce simple tracé de la courbe, qui, à l'œil nu, n'est pas facilement interprétable du point de vue du nombre de battements par minute.

Nous avons donc créé un algorithme qui permet de fixer deux seuils : un seuil inférieur et un seuil supérieur. Celui-ci permet que, dès que la valeur à l'instant t du signal est supérieure à $Seuil_{inf}$, et que cette valeur est inférieure à $Seuil_{sup}$, alors on calcule le temps entre deux pics grâce à la fonction $millis()$ présente par défaut dans le compilateur Arduino.
Illustrons ce raisonnent par un schéma. L'objectif de la définition de ces seuils inférieurs et supérieurs est de concentrer notre analyse du signal sur les pics car c'est bien l'écart temporel entre ces derniers qui nous intéresse pour pouvoir calculer la fréquence cardiaque (Hz) et ainsi le rythme cardiaque en nombre de battements par minute (BPM).



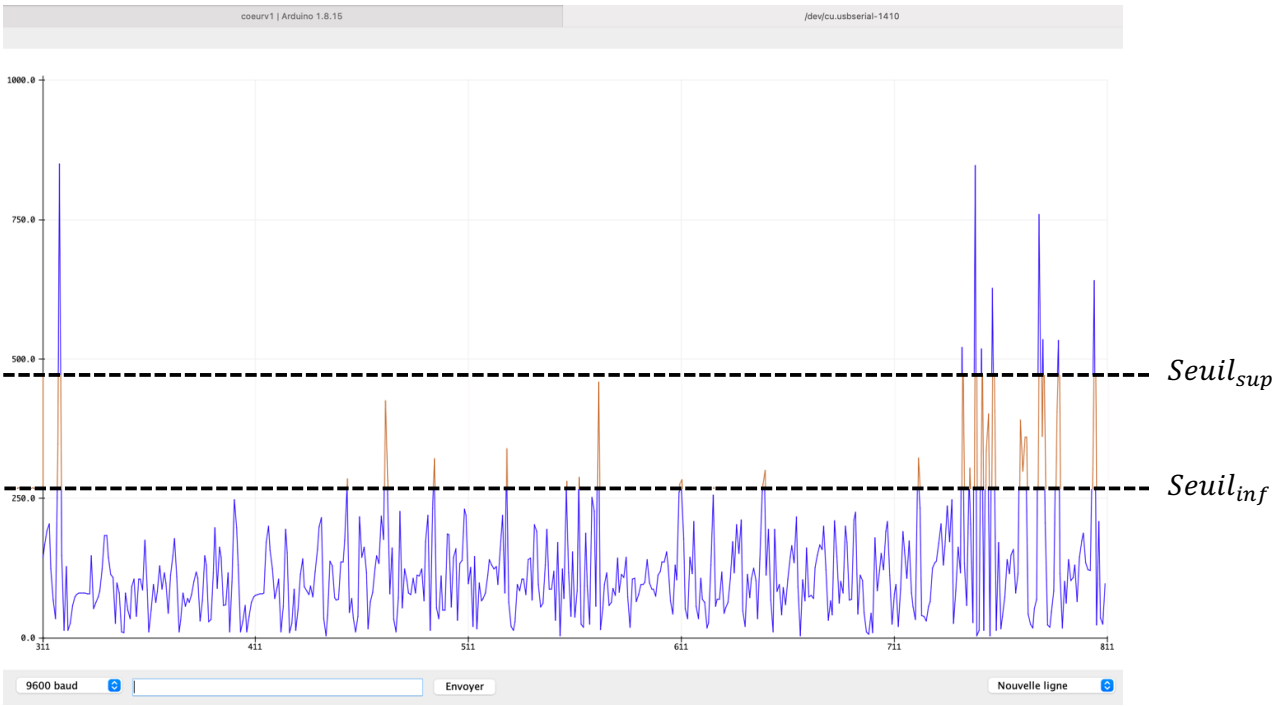


Le montage ci-dessus illustre bien ce que nous souhaitons analyser de par l'utilisation de $Seuil_{inf}$ et de $Seuil_{sup}$. La figure de gauche correspond à une capture d'écran que nous avons effectué sur l'oscilloscope. Elle représente le signal amplifié et filtré donc notre rythme cardiaque. On peut manuellement calculer notre fréquence cardiaque (en Hz puis en BPM) grâce aux curseurs de l'oscilloscope. Mais cette manipulation est longue et fastidieuse. C'est pourquoi l'utilisation d'un algorithme semble très adaptée.
Néanmoins, l'algorithme n'est pas capable de repérer tels quels les pics du signal car ce dernier est constitué de différents pics, certes d'amplitudes différentes mais il demeure néanmoins inexploitable à ce stade.

C'est pourquoi l'utilisation de $Seuil_{inf}$ et de $Seuil_{sup}$ permet de concentrer l'analyse sur des pics d'amplitude comprise dans une même plage de tensions. Ici, nous focalisons le traitement sur les pics en orange représentés sur la figure de droite ci-dessus. Cela permet de ne pas perturber notre algorithme avec des pics d'amplitude inférieure (qui correspondent à la partie du signal en verte qui sont donc éliminés grâce à l'utilisation de $Seuil_{inf}$) et des pics d'amplitude supérieure correspondant généralement à un mouvement du doigt sur le capteur (éliminés grâce à l'utilisation de $Seuil_{sup}$).

Pour l'ajustement de $Seuil_{inf}$ et de $Seuil_{sup}$, il faut visualiser le signal issu de la lecture du pin A_0 de l'Arduino sur le moniteur série. On repère alors approximativement avec une série de tests (sur plusieurs personnes) et d'ajustements les valeurs assignées à $Seuil_{inf}$ et $Seuil_{sup}$, comme on peut l'observer sur la courbe ci-dessous. Attention néanmoins, ici les échelles en abscisse et en ordonnée sont respectivement en ms et codé sur 255 en ordonnée.

On a donc fixé $Seuil_{inf} = 300$ et $Seuil_{sup} = 460$.

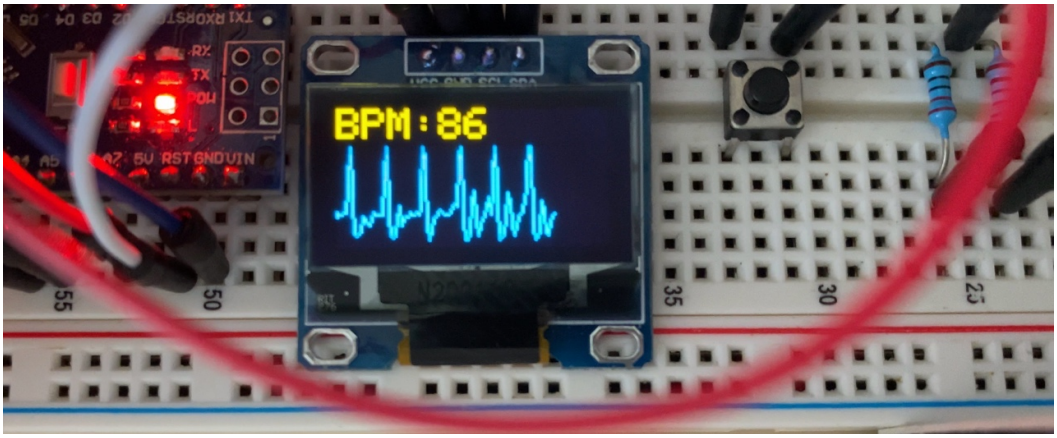


Nous avons également ajouté différentes fonctionnalités supplémentaires associées à la programmation :

⇨ L'affichage d'un message au démarrage de l'oxymètre et tant que le doigt n'est pas positionné sur le capteur, des instructions s'affichent sur l'écran.

⇨ L'ajout d'une LED RGB, qui s'allume et s'éteint brièvement à chaque pulsation cardiaque (ce qui correspond aux pics compris entre $Seuil_{inf}$ et $Seuil_{sup}$ décrits précédemment).
- Si $50 \leq BPM \leq 140$ alors la LED clignote à chaque impulsion cardiaque en vert.
- Si $BPM < 50$ alors la LED clignote à chaque impulsion cardiaque en rouge.
- Si $BPM > 140$ alors la LED clignote à chaque impulsion cardiaque en rouge.

⇨ L'affichage du rythme cardiaque en BPM et de le tracé de la courbe sur écran OLED.
Vidéo disponible à l'adresse :
https://drive.google.com/file/d/1MraWeqGs4y1higITqW9UQLiGy16kPw_I/view?usp=sharing



La communication du signal, du nombre de battements par minute ainsi que des informations complémentaires se fera par le biais de l'écran OLED (qui sera clipsé sur les connecteurs soudés au PCB) ainsi que par l'application mobile. Détaillons maintenant les fonctionnalités de cette dernière.

Remarque :
Le programme que nous avons codé sur l'IDE Arduino sera disponible sur la page WEB de l'atelier que nous avons créé à l'adresse suivante :

https://perso.esiee.fr/~ferracaa/projets/OxymetriePage/oxypage1.html

Application mobile

Un message est diffusé sur l'écran au démarrage de l'oxymètre pour rappeler à l'utilisateur que l'application renferme des fonctionnalités complémentaires. La LED tricolore s'allume d'ailleurs en jaune.

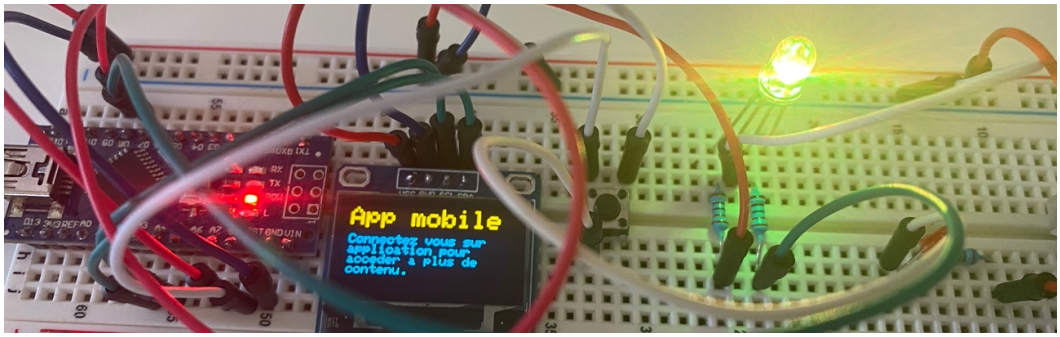


Figure 13 : Message sur écran concernant l'application mobile et LED

Nous avons décidé de mettre en place une application mobile afin de permettre à l'utilisateur de visualiser son pouls en temps réel. Cette application a été développer sur le site internet suivant :

https://appinventor.mit.edu/

Nous nous sommes connectés grâce à un compte mail notamment ici celui de l'ESIEE.

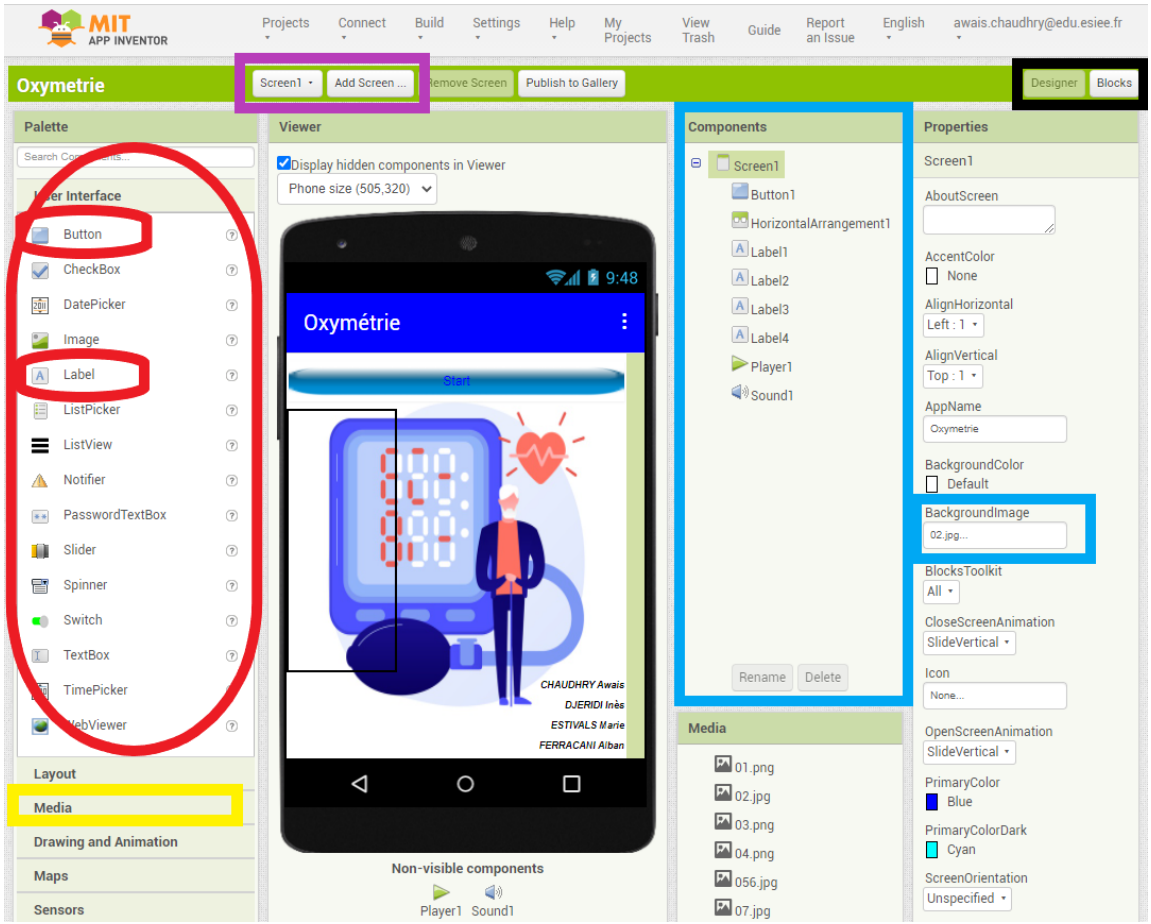


Figure 14 : Interface internet de MIT AppInventor

La partie en haut, en violet, nous permet de changer ou d'ajouter un écran (cette fonctionnalité est utile par exemple lorsque nous ajoutons un bouton et que nous voulons que suite à notre appui on change d'écran).

La partie à gauche, en rouge, va nous permettre d'ajouter un bouton ou encore du texte sur notre écran. Par exemple ici le bouton « start » et nos identités sur l'écran d'accueil. Les propriétés du bouton et du texte se font à droite (la partie en bleu).

La partie à droit, en bleu, on choisit le composant que l'on souhaite modifier puis on modifie au niveau de « Properties ». Par exemple ici nous avons choisis le composant « Screen1 » et nous lui avons fixé un fond d'écran qui est l'image « 02.jpg ».
Pour accéder à cette image nous avons dû la charger au niveau de « Media » :

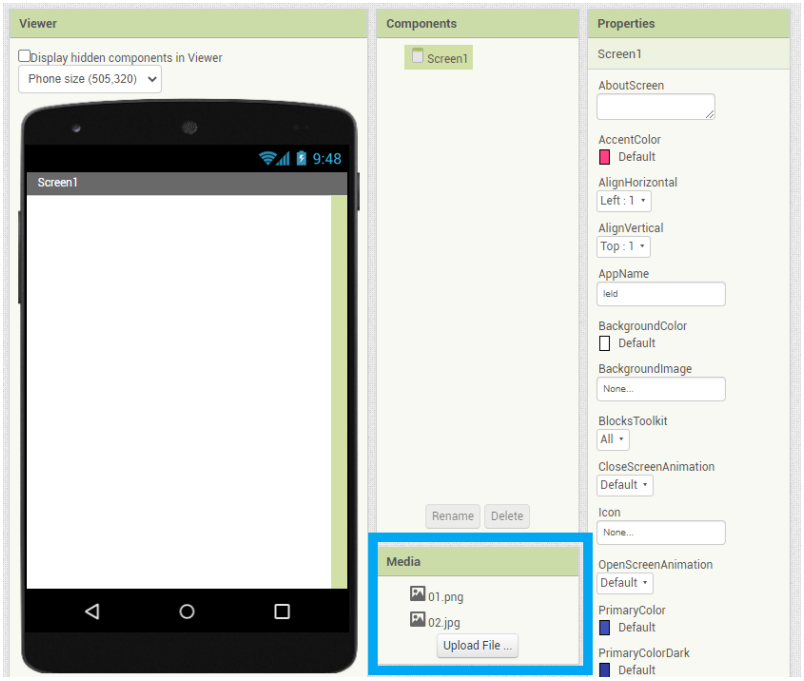


Figure 15 : Chargement d'un document sur le site internet de MIT AppInventor

Une fois l'image chargé nous pouvons l'utiliser comme fond d'écran ou encore comme icône d'un bouton.

Dans la figure 1, à gauche en jaune au niveau de « media » nous pouvons ajouter des images, de la musique ou encore des sons sur notre écran.
Par exemple nous pouvons ajouter un son à chaque fois qu'on appuie sur un bouton ou encore nous pouvons mettre des images pour rendre l'écran plus attractif comme dans l'exemple ci-dessous :

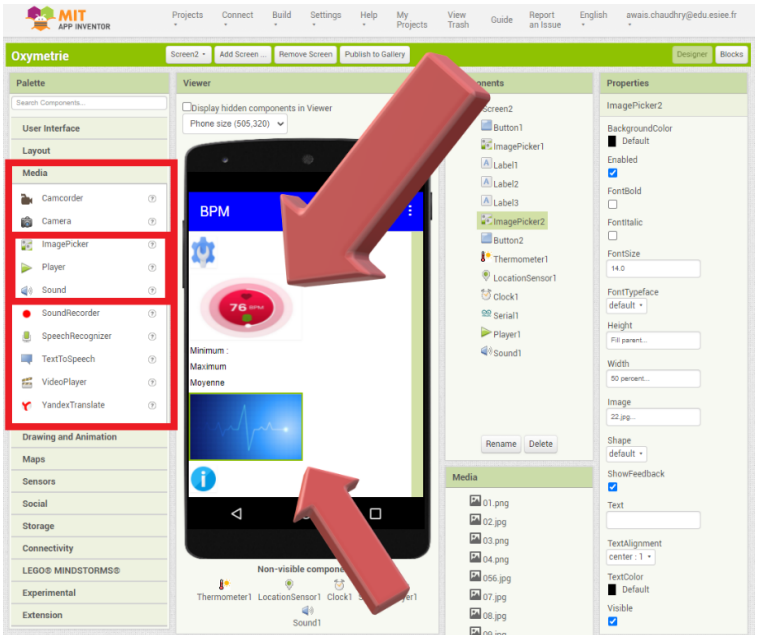


Figure 16 : Mise en place d'image sur un écran depuis le site internet de MIT AppInventor

Enfin pour que la musique fonctionne sur un écran ou encore pour qu'un bouton fonctionne il faut un programme derrière tout cela. C'est pourquoi nous allons nous intéresser maintenant à la partie programmation.
Au niveau de la figure 1 nous pouvoir voir en haut à droite en noire que nous avons deux interfaces. L'une est « Designer » elle s'occupe du côté esthétique en ajoutant l'emplacement des boutons, le fond d'écran,

le texte etc… et l'autre « Blocks » s'occupe du fonctionnement par exemple que se passe-t-il suite à l'appui d'un bouton ou lorsqu'un nouvel écran s'ouvre.

Par exemple prenons le cas de l'écran d'accueil, nous voulons que lorsque cet écran s'ouvre alors une musique démarre, que lorsque l'on clique sur le bouton « start » qui ici se nomme « Button1 » cela arrête la musique et que ça produit le son spécifique que nous avons chargé au préalable et que cela nous fasse changer d'écran suite au clique.

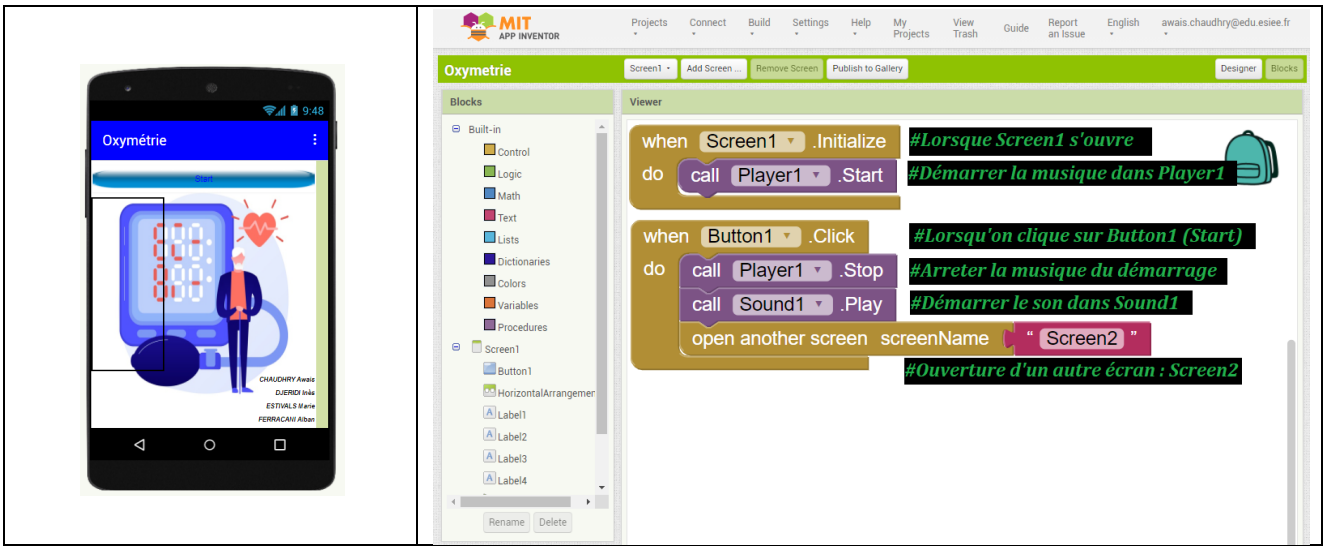
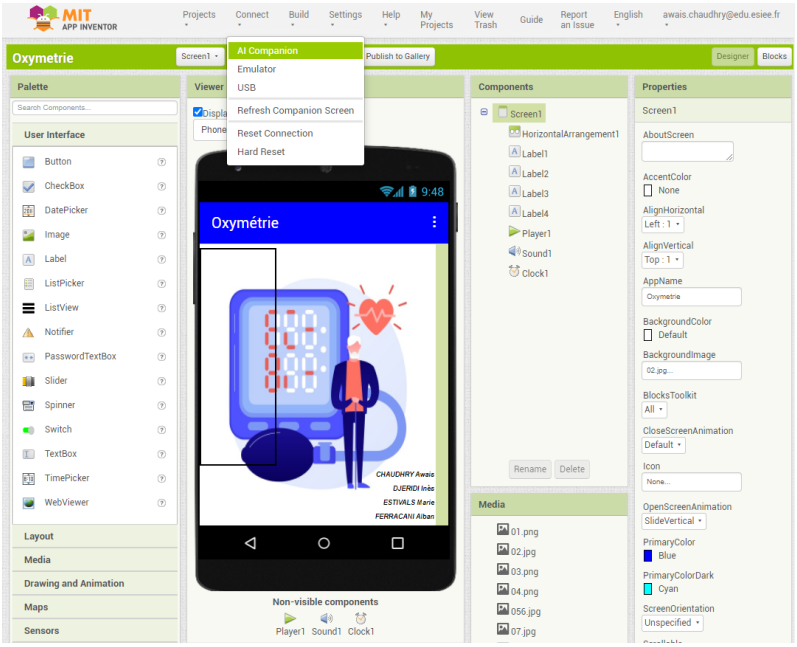


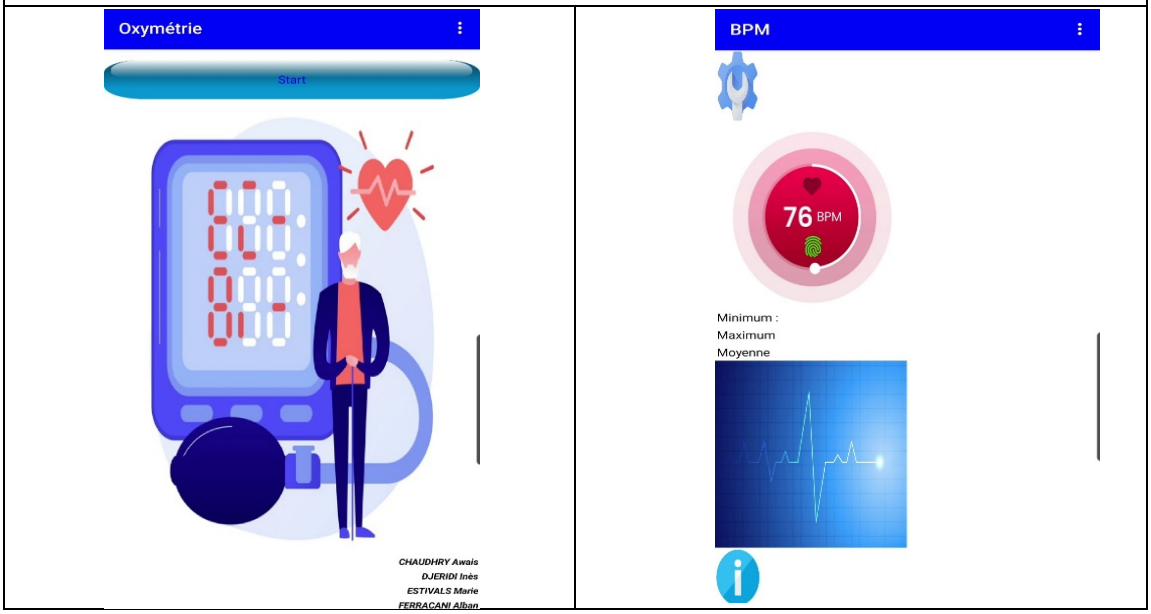
Figure 17 : Partie programmation sur site internet de MIT AppInventor

Pour pouvoir utiliser l'application sur son téléphone il faut se rendre sur le site internet et générer un QR code qu'il faudra scanner en se rendant sur l'application MIT AppInventor sur son smartphone :

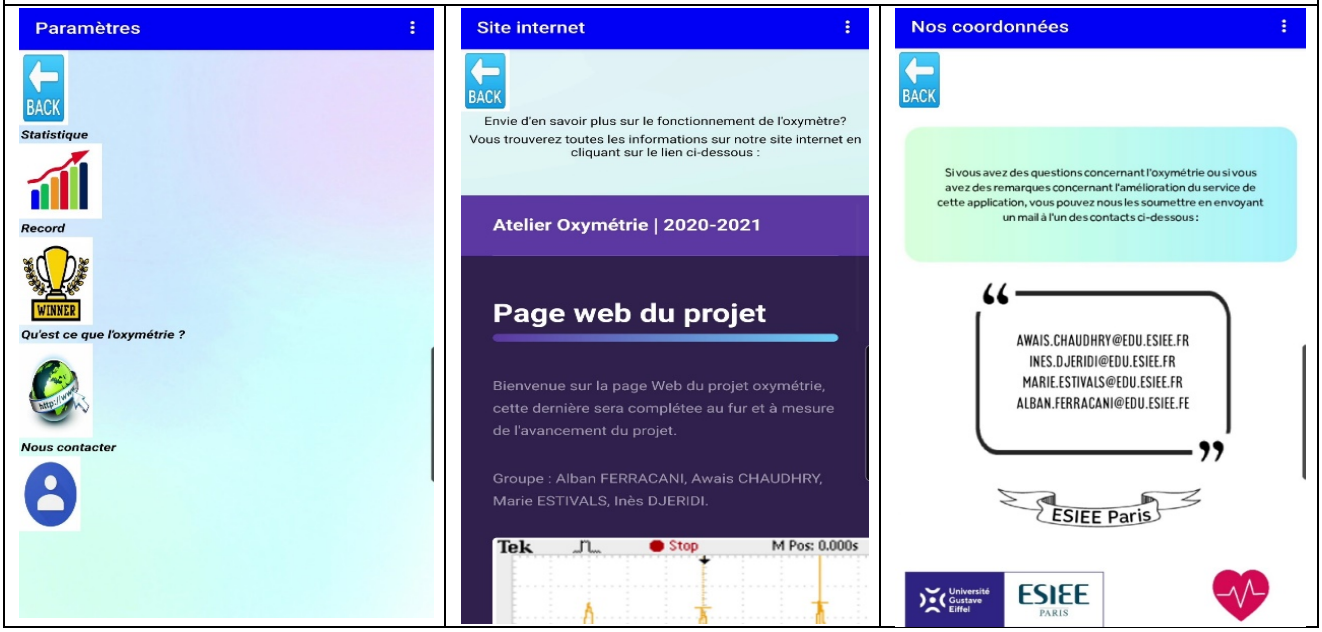


Si nous n'avons plus aucun changement à faire sur l'application il existe une autre solution qui est plus pratique, qui consiste à télécharger notre propre application sur notre smartphone en générant un fichier type apk.

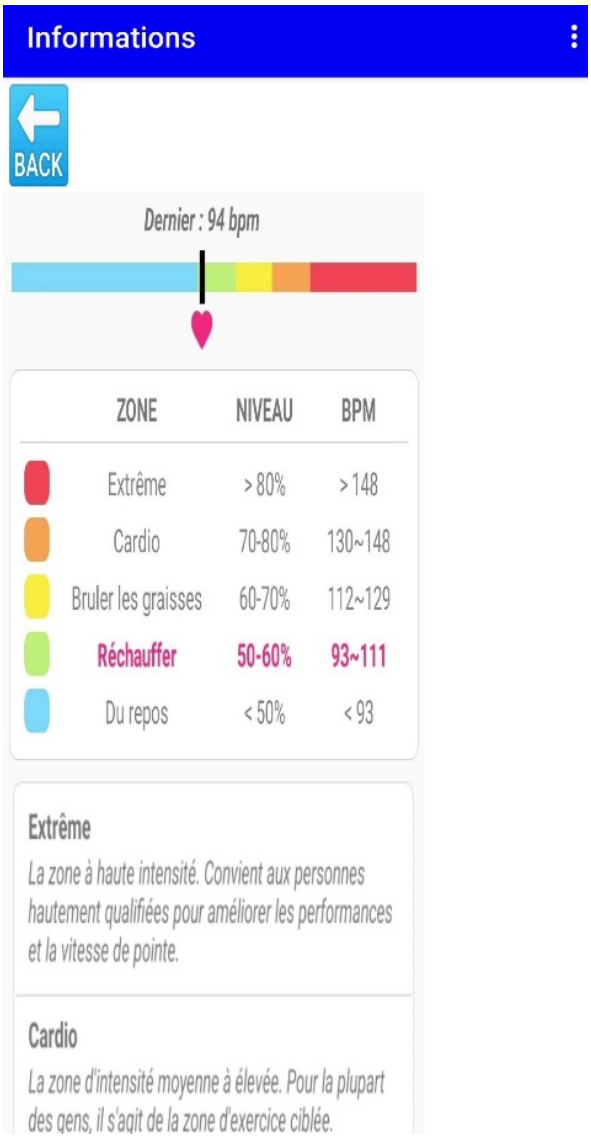
Ecran d'accueil



Paramètres



Il nous manque les parties « Statistique » ainsi que « Records » car pour l'instant il n'y a pas encore eu d'études.

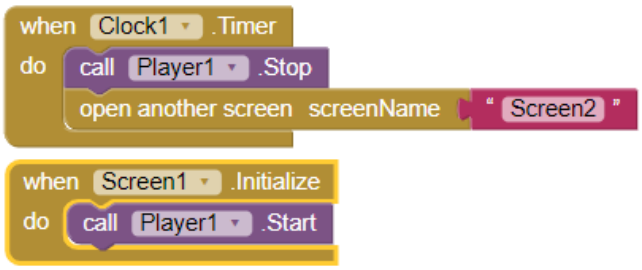


Nous avons également mis en place un bouton d'information au niveau de l'écran d'accueil (là où sera affiché le pouls de l'utilisateur) afin qu'il puisse accéder à la page ci-contre et analyser selon la situation dans laquelle il se trouve si son pouls est correct ou non.

Les images et les écritures peuvent être déformées selon la taille de l'écran sur laquelle l'application est utilisée.

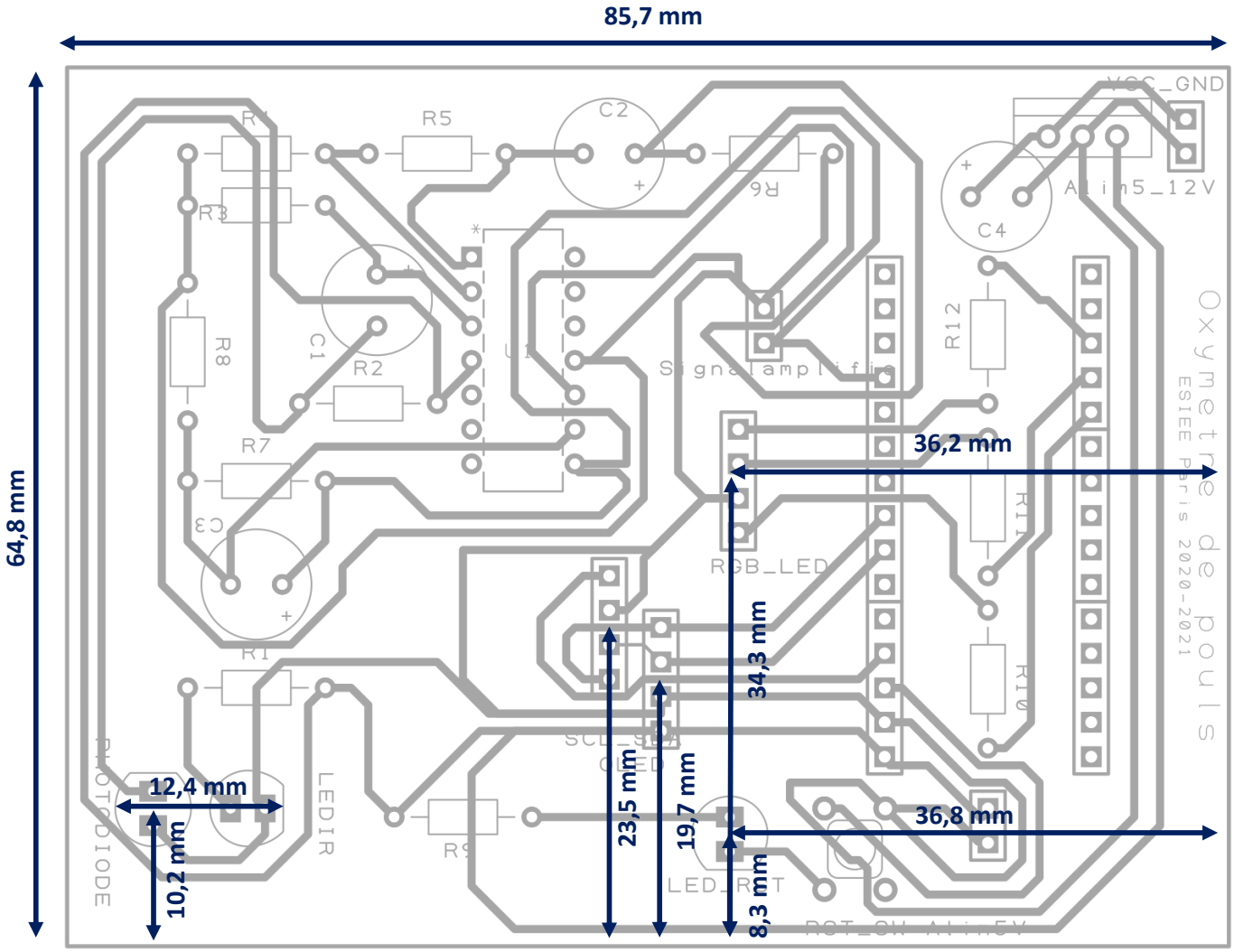
Enfin nous avons modifié notre page d'accueil en supprimant le bouton « Start » que nous avons remplacé par un compte à rebord de 10 secondes au bout duquel l'utilisateur est automatiquement redirigé vers le deuxième écran d'accueil sur laquelle la mesure de son pouls pourra être effectuée.

La partie du code de la figure 17 a donc, par conséquent, été modifiée.



Modélisation d'un boitier 3D

Rappelons d'abord quelques dimensions utiles à la modélisation du boitier en 3D. Ici, nous utiliserons le logiciel FUSION 360 et nous exporterons le fichier au format stl.



A. Esquisse du boitier (coque extérieure)

Le boitier sera constitué de plusieurs éléments, assemblés les uns avec les autres :

- ⇨ Une coque extérieure adaptée aux dimensions du PCB. Le PCB viendra y être vissé et isolé par l'intermédiaire d'entretoises.

- ⇨ Un capot supérieur permettant d'accueillir un support de doigt pour l'oxymètre. Le capot sera creux en partie afin de permettre la vision de l'écran ainsi que des LED et pour faciliter l'accès au bouton poussoir. Cela permettra également

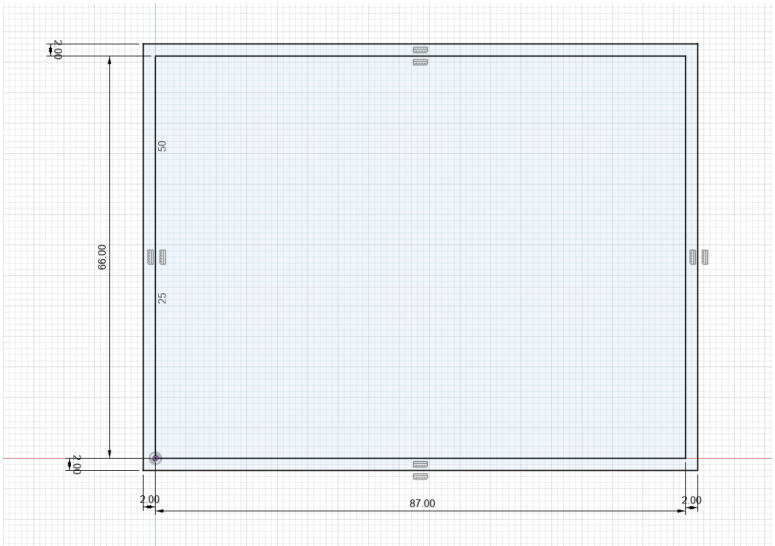


Figure 18 : Dimensions extérieures du boitier extérieur

d’assurer une régulation thermique du boitier, qui, de par l’utilisation d’un régulateur de tension peut être amené à chauffer en cas d’utilisation prolongée.

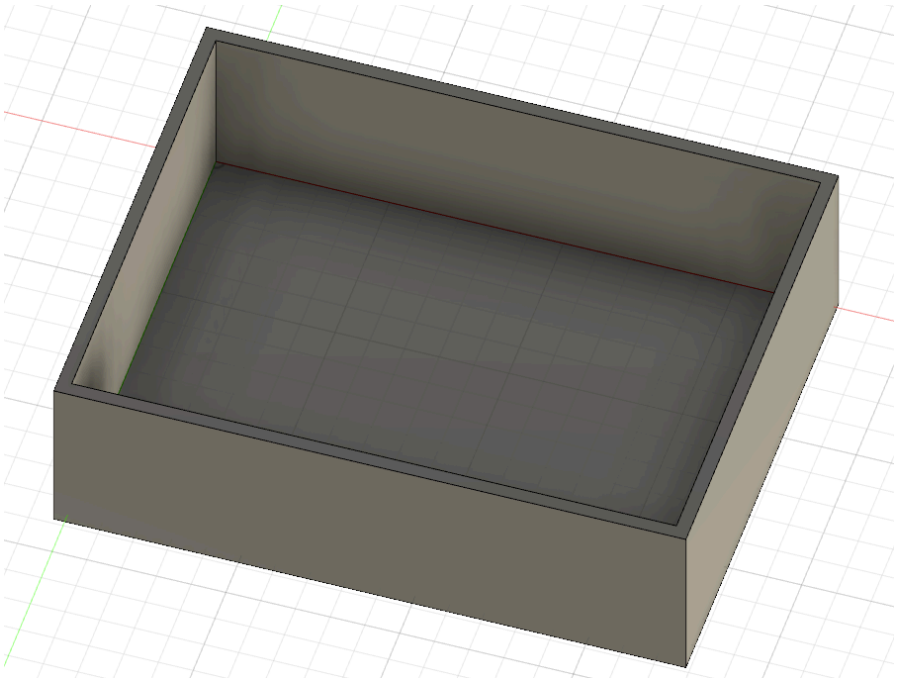


Figure 19 : Vue 3D de la coque extérieure

B. Esquisse du capot (coque supérieure + support doigt)

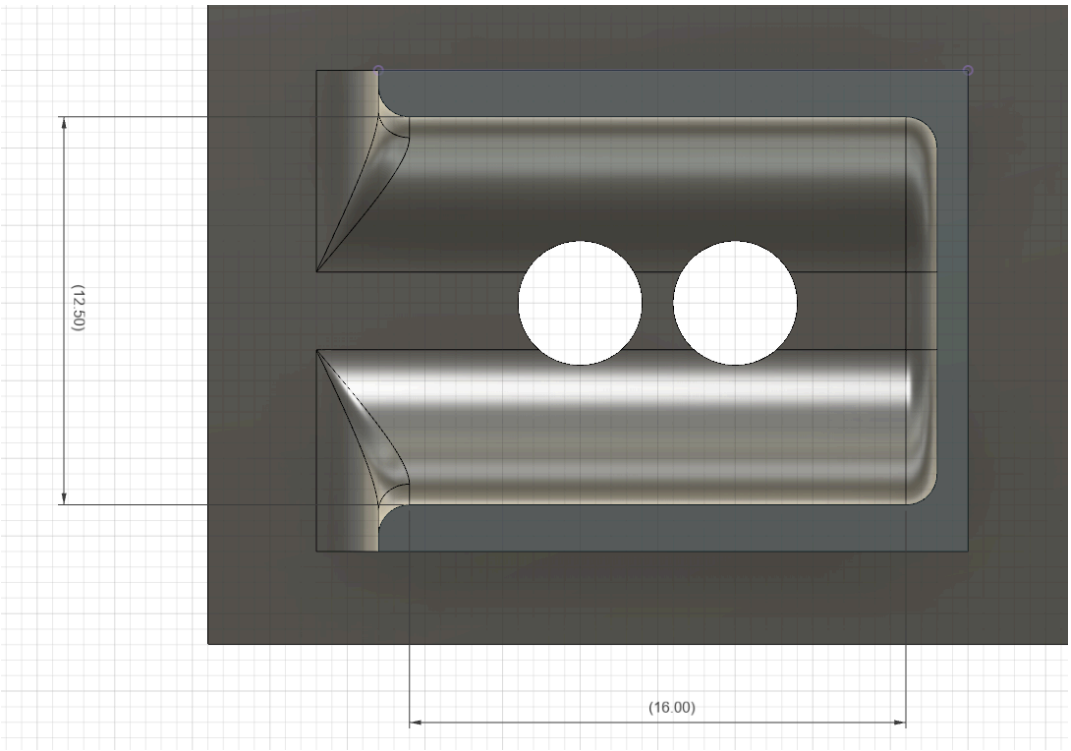


Figure 20 : Support de doigt afin de ne pas bouger pendant la mesure

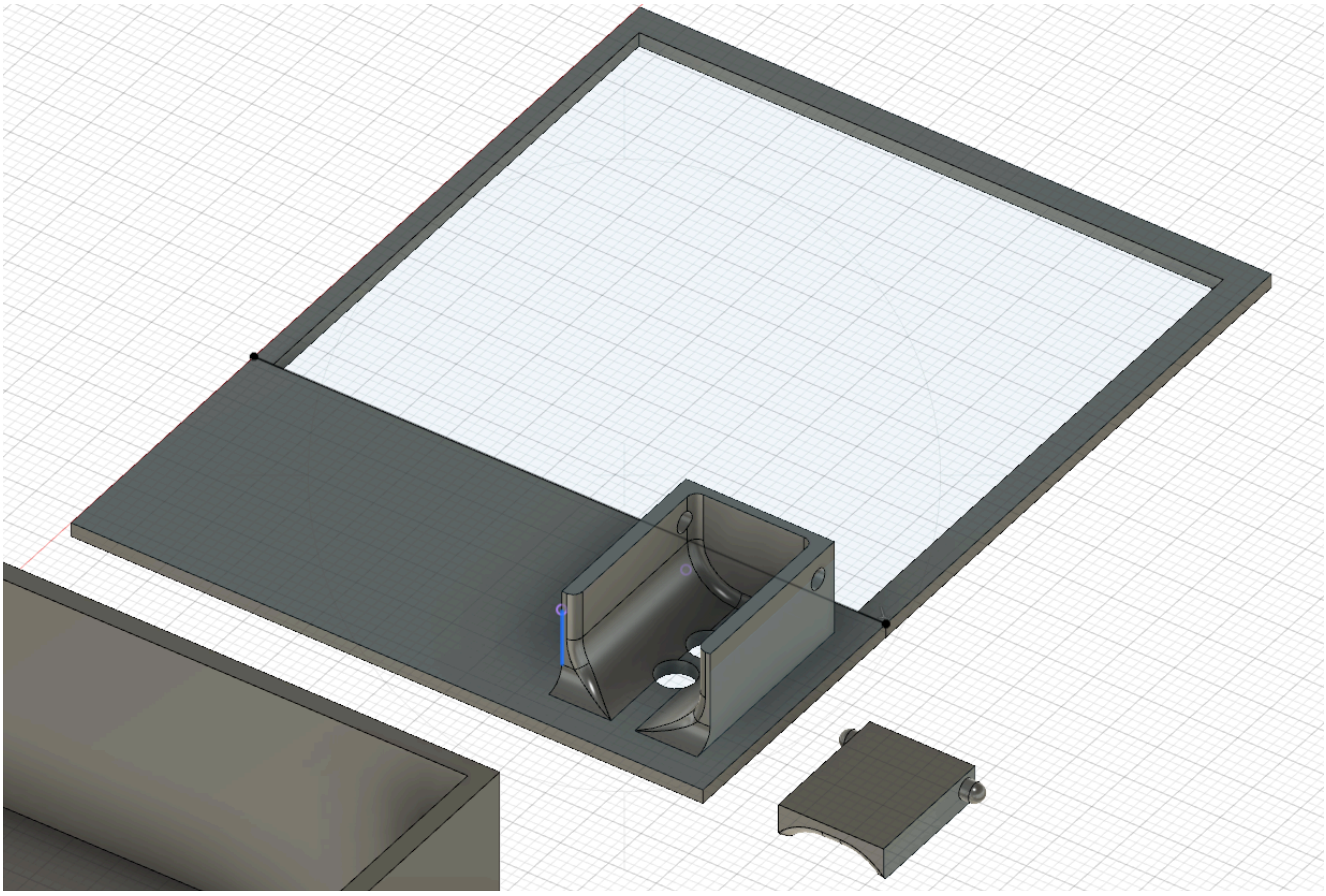


Figure 21 : Vue 3D du capot supérieur avec support de doigt

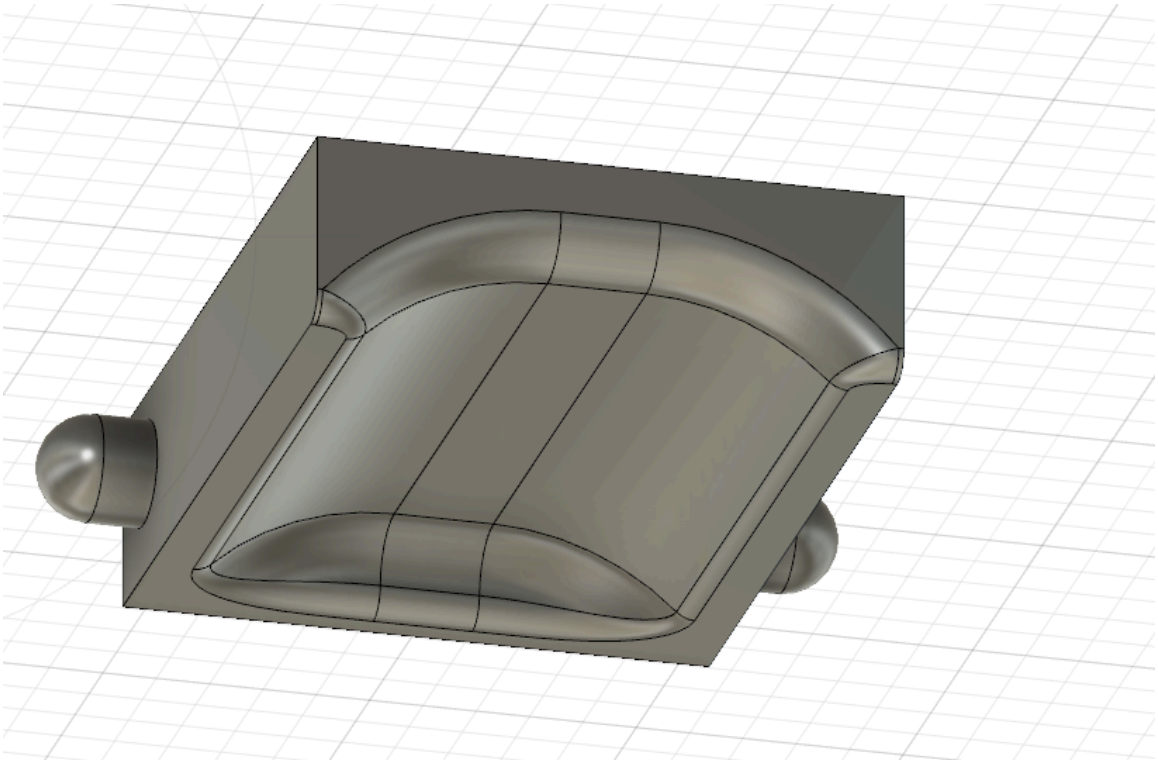
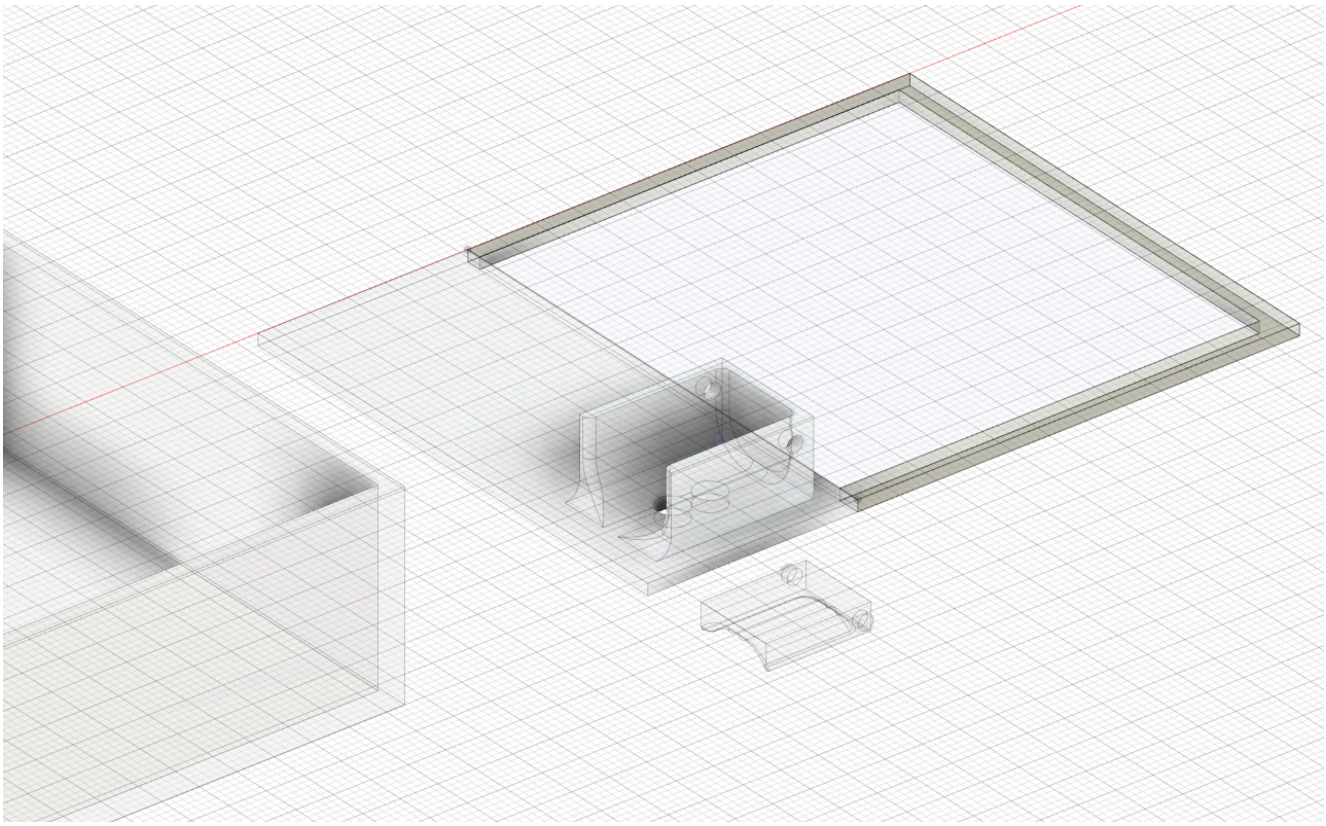


Figure 22 : Pince supérieur venant tenir le doigt en position sur le support qui sera clipsée dessus

C. Protocole d'impression 3D



Pour faciliter l'impression 3D et l'accrochage de l'impression au plateau de l'imprimante, toutes les pièces (au nombre de 3) ont une face entièrement plane.

Remarque :
Nous avons mis en place un lien permettant de voir la conception 3D du boitier directement en ligne et ainsi de pouvoir le regarder sous tous ses angles.

- ⇨ Lien : https://a360.co/3cw7WFn
- ⇨ Mot de passe : ESIEE2021

Datasheet des composants et informations utiles relatives à la conception et au dimensionnement du PCB.

Arduino Nano

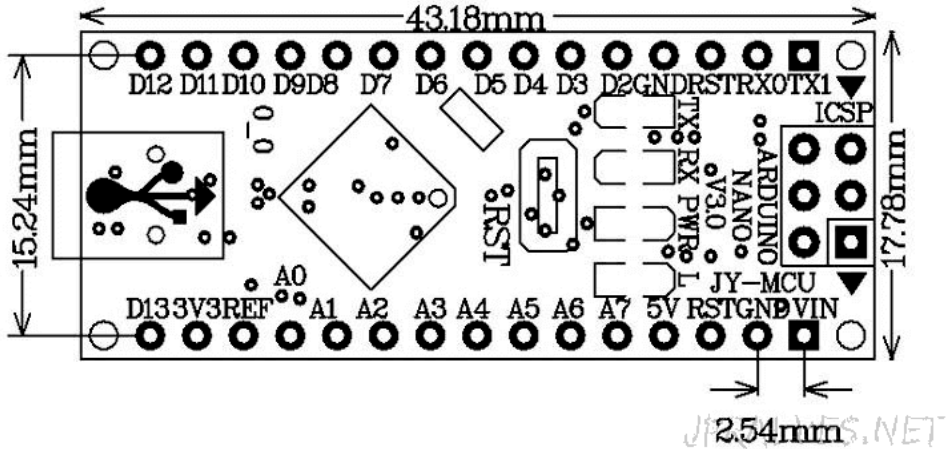


Figure 23 : Dimensions et pinout d'un Arduino Nano
Source : https://urlz.fr/fRFQ

Bouton poussoir

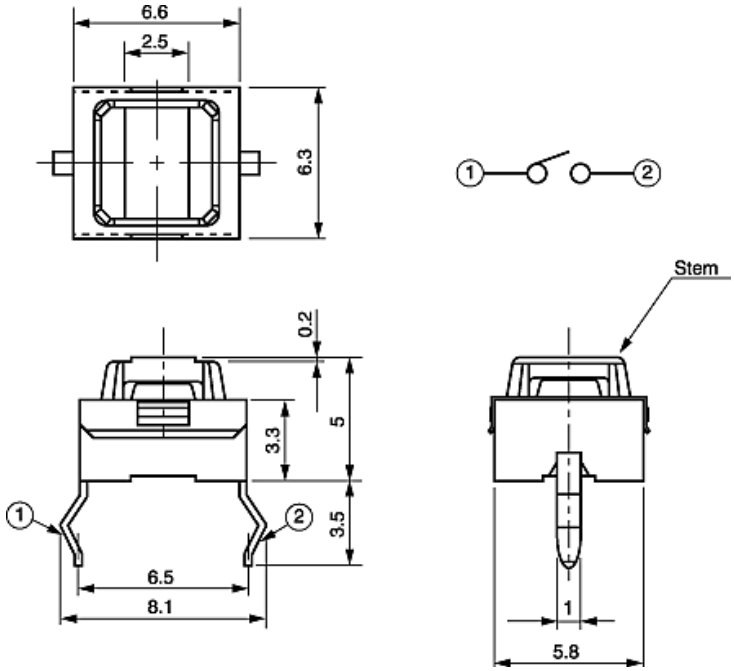


Figure 24 : Dimensions et pinout bouton poussoir (reset dans notre cas)
Source : https://urlz.fr/fRFR

LED RGB

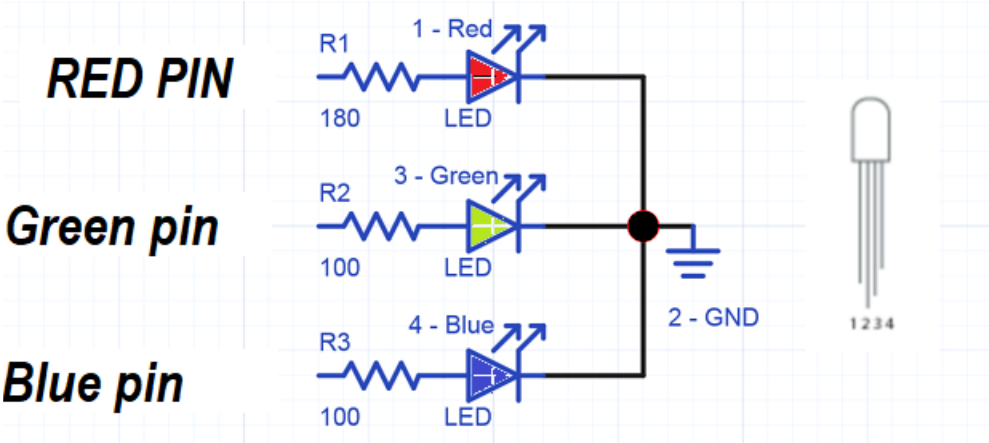


Figure 25 : Dimensions et pinout led RGB

Source : https://urlz.fr/fRFT

Régulateur de tension LM7805

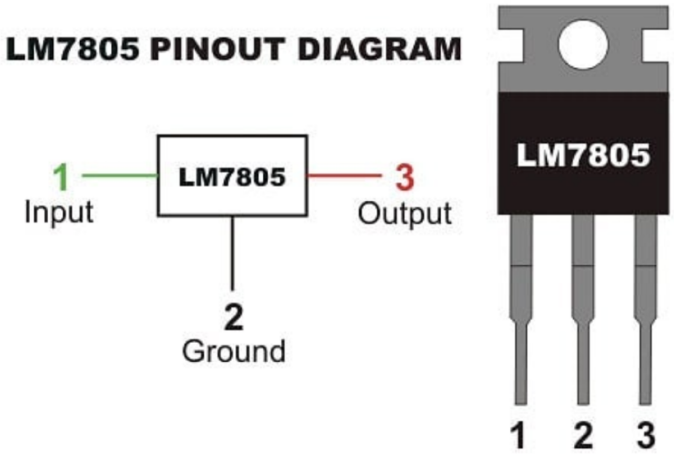


Figure 26 : Datasheet régulateur de tension LM7805

Source : https://urlz.fr/fRFV

BLE P CLICK (BLUETOOTH SHIELD)

https://download.mikroe.com/documents/add-on-boards/click/ble-p/ble-p-click-schematic-v100.pdf

https://www.mikroe.com/blog/bluetooth-low-energy-part-1-introduction-ble

https://www.mikroe.com/blog/bluetooth-low-energy-part-2-ble-p-click

https://github.com/MikroElektronika/Click_BLE_P_NRF8001