Lycée
LA FAYETTE
Clermont-Ferrand

CENTRALE DE MESURE AUTONOME POUR PISCINE

Dossier Commun

Projet BTS SN 2023

COSTES Nathan - POTON Thomas - FRADIER Florent -HADDADOU Tommy -
NDINGA Christian - DUZELIER Théo

Sommaire :

1. Présentation de l'entreprise « donneur d'ordre »

L'entreprise « donneur d'ordre » est le lycée LA FAYETTE. Cette centrale de mesure autonome s'adresse à un particulier souhaitant visualiser la température, le PH, le redox ainsi que la pression et ainsi pouvoir contrôler l'état de sa piscine.

2. Présentation rapide des membres de l'équipe

Nous sommes au totale 6 personnes dans l'équipe.

Il y a 4 personnes (IR) qui doivent traiter les données puis créer un affichage graphique sur différents systèmes (Tommy, Florent, Nathan et Thomas).

Dans la partie physique (carte électronique et programme Arduino) (Theo et Christian), on doit traiter la récéption des données.

3. Définition du besoin exprimé par l'entreprise

a. Mission principale du système.

Mon système est nécessaire car l'entretien d'une piscine est très important, d'un point de vue sanitaire comme esthétique.

Il doit être fiable sans quoi des problèmes de santé et de peau peuvent survenir.

Le but du système est que l'utilisateur peut contrôler l'état de son eau et ainsi pouvoir régler le taux de PH (redox, pression…) sans que celui-ci ne soit si trop acide ni trop basique.

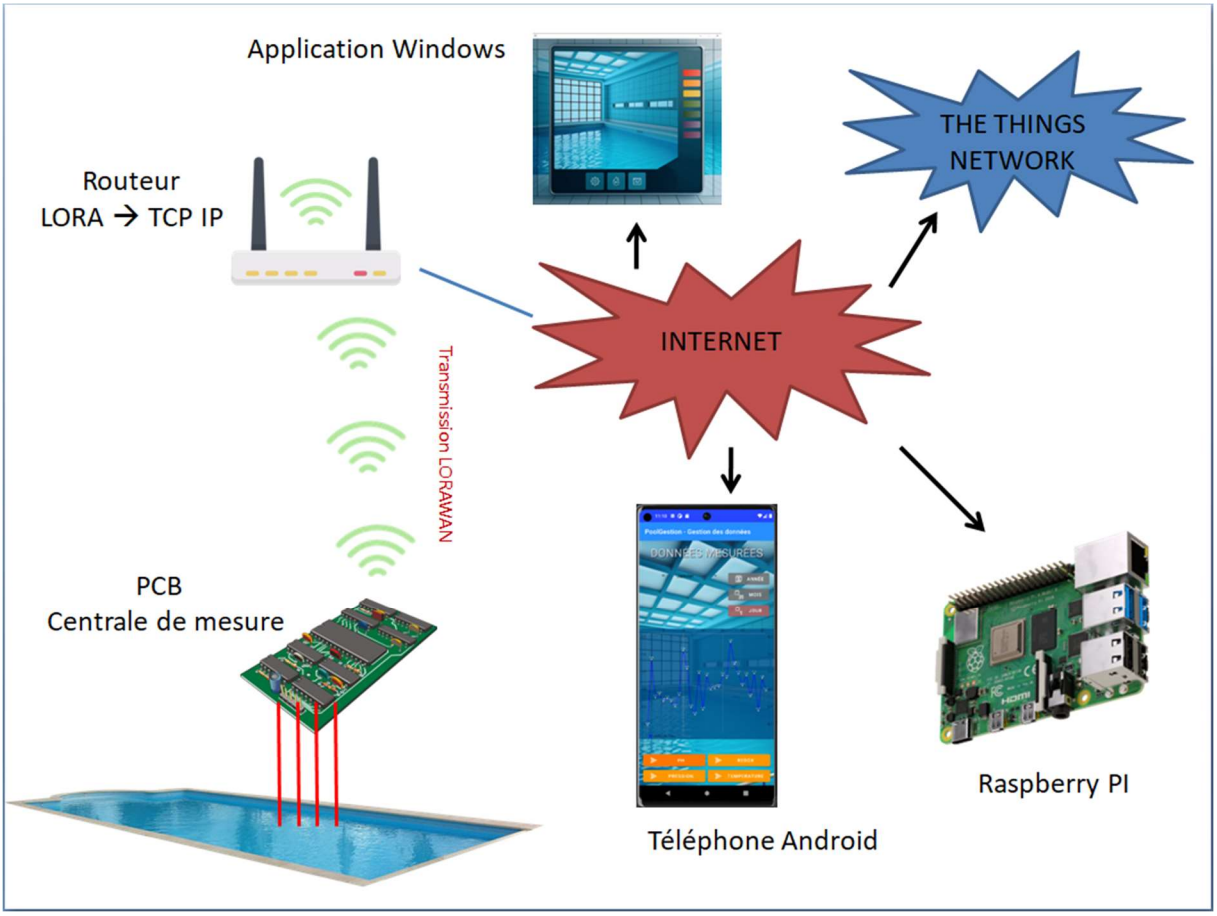
b. Mise en situation du projet dans son contexte industriel

Il y a déjà des systèmes équivalents qui existe sur le marché, or ce sont des matériels qui sont couteux. En ayant une bonne optimisation des composants nous pouvons peut-être réduire son prix.

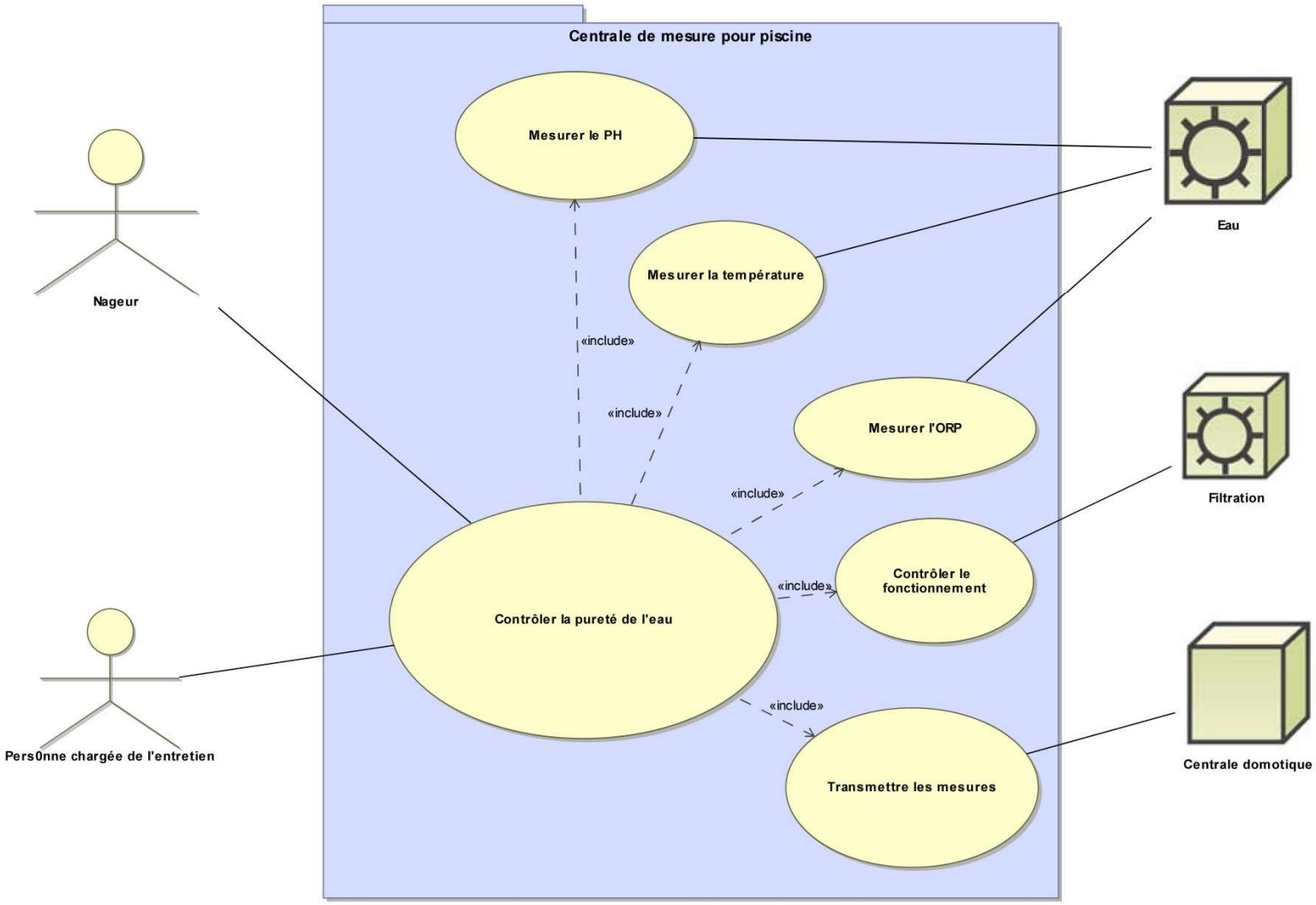
Le marché visé : ce produit rend service à un particulier possédant une piscine et souhaitant contrôler l'état de son eau.

L'utilisateur finale sera bien évidemment l'acheteur

c. Contexte du système (Diagramme de contexte, description textuelle)



d. Utilisation (Diagramme des cas d'utilisation)



4. Dossier présentation du système technique.

Une piscine doit conserver un équilibre chimique particulier, sans quoi l'eau sera moins pure, irritante, colorée, le matériel se détériorera.

L’analyse de l’eau est une étape indispensable dans l’entretien d’une piscine. Il s’agit d’équilibrer son pH, mais aussi d’ajuster son traitement.

La centrale de mesure pour piscine permet la mesure du PH, du Redox ou ORP (Oxydo Reduction Potential), et de la température de l’eau, elle contrôle aussi le fonctionnement correct de la pompe de filtration. Les informations collectées sont envoyées à une centrale domotique permettant l’affichage et la gestion des paramètres.

Le pH ou potentiel d’hydrogène est lié à la concentration en ions H+ qui représentent l’acidité de l’eau. L’échelle de pH s’étend de 0 (milieu très acide : concentration d’ions H+ importante) à 14 (milieu très basique : faible concentration d’ions H+). Un pH neutre est équivalent à 7. En piscine, le pH idéal se situe entre 7 et 7.2.

Le Redox ou ORP (Oxydo Reduction Potential) indique le pouvoir oxydant ou réducteur d'une substance. Il permet de juger de l'aspect général de l'eau d'un point de vue sanitaire. L'ORP se mesure en millivolts.

La température idéale en piscine doit se situer à environ 10 °Celsius en dessous de la température du corps soit environ 27°C. La température idéale pour profiter au maximum de sa piscine extérieure est donc comprise entre 26 et 28°C.

La filtration est l’action mécanique qui permet de faire circuler l’eau de la piscine, de retenir au travers du filtre les particules en suspension et d’assurer la bonne diffusion du désinfectant.

La centrale de mesure pour piscine doit transmettre les résultats de différentes mesures vers une passerelle LoraWAN afin d’intégrer les mesures dans les données domotiques de l’habitation.

La base de données utilisées pour les applications et interfaces est basée sur un Raspberry.
Celui-ci fonctionne avec le système d'exploitation Linux ou est utilisée la base données grâce à un serveur MariaDB.

La base de données reçoit les données de la passerelle LORAWAN. Un programme permet de convertir les données reçues en format JSON vers un format SQL.

Le base de données et le Raspberry peuvent être configurés localement mais aussi via une page Web permettant aussi de consulter les données sous forme de graphiques comme l'interface Windows et Android.
Localement, est présent un affichage de données numériques.

5. Analyse des exigences techniques, mise en forme du cahier des charges.

Utilisation d'une sonde de mesure de Ph « Electrode pH ASP200-2-1M », la précision devra être inférieure à 5% et la résolution de 0.1.

La mesure du Redox ou ORP (Oxydo Reduction Potential) est réalisé par une sonde ORP 110020293. La résolution sera 10mV.

La mesure de température est réalisée par une sonde DS1820, résolution 0.5°C, précision 0,5%.

Contrôle du fonctionnement de la pompe de filtration par capteur de pression SEN0257 (0 à 1MPa), résolution et précision 500Pa.

Transmission des données vers une passerelle LoraWAN.

Conception d'une application Windows permettant de configurer le système et d'afficher les paramètres mesurés de la centrale de mesure en temps réel.

Création d'un système de configuration embarqué sur la carte serveur Raspberry permettant de configurer la carte et d'accéder aux mesures.
Tous les affichages se feront dans une IHM sur une page web. Ce site devra être compatible avec une majorité de navigateur web.

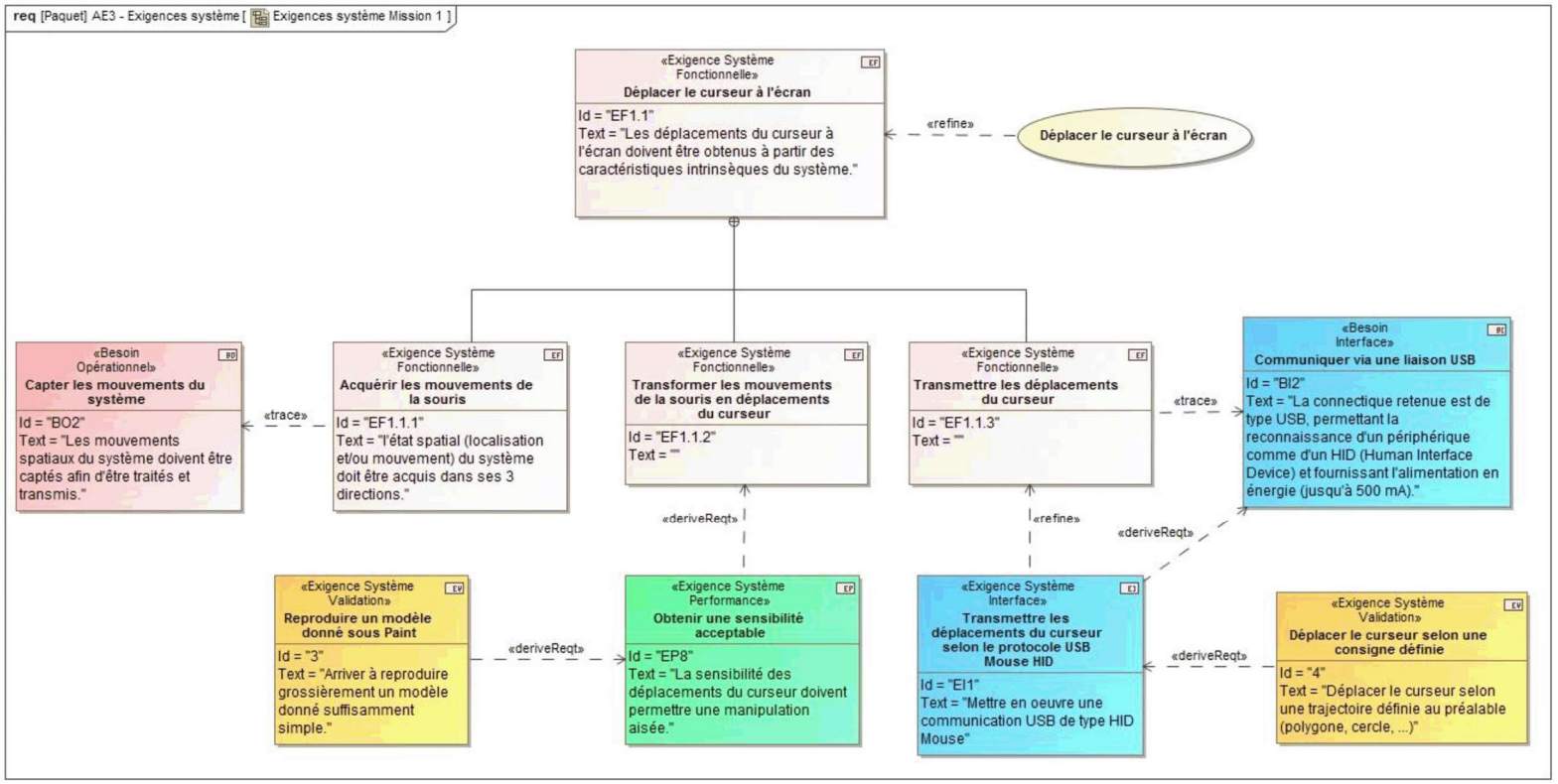
Créer une base de données, permettant d'enregistrer les valeurs mesurées (1an de sauvegarde) avec une détection d'alerte en cas de problèmes sur la carte Serveur Raspberry pi.
Lorsqu'un problème est détecté une alerte sonore et lumineuse sera émise par la carte raspberry.
Une application Windows sera développée pour afficher les courbes.
Conception d'une application Android permettant de configurer le système et d'afficher les paramètres mesurés en temps réel.

Planning :

				V	V		R							V	V	R	EX		
Repère Activité	DESCRIPTION DE LA TÂCHE	DUREE	05	06	07	08	09	10	11	12	13	14	15	16	17	18	19	20	21
A1	Présentation de l'entreprise	1	1																
A2	Justification du besoin	1	1																
A3	Elaboration du cahier des charges préliminaire	3	3																
A4	Mise en forme du cahier des charges en fonction des choix technologiques	3	3																
A5	Inventaire des solutions technologique permettant de répondre au CDC	6	5			1													
A6	Validation du CDC et des choix technologiques par l'entreprise	2				2													
A7	Elaboration du planning prévisionnel	4				4													
A8	Elaboration du cahier de recette préliminaire	4				2	2												
	Rédaction de la partie commune du rapport	/																	
	Revue de projet N°2	(24)					X												
A9	Elaboration des diagrammes structuraux SysML	6					6												
A10	Présentation du CDC et fonctionnelle de la partie individuelle	6					5	1											
A11	Choix technologiques de composants, structures et solution	10						7											
A12	Tests préliminaires sur maquette d'expérimentation	2						2											
A13	Elaboration des schémas fonctionnels des parties individuelles	3							3										
A14	Elaboration d'une carte électronique	36							6	13	9	5							
A15	Mise à jour du planning et chiffrage	2										2							
A16	Elaboration des algorithmes généraux	2										2							
A17	Rédaction d'un cahier de recette préliminaire	24										4			9	11			
	Revue de projet N°3	(101)														X			
A18	Chiffrage détaillé	4														2		2	
A19	Elaboration d'une fiche de maintenance curative.	10																10	
A20	Elaboration d'une fiche de maintenance préventive	4																1	2
A21	Rédaction du cahier de recette	6																	1
A22	Bilan	4																	1
	Rédaction du rapport individuel	/																	
	Total	129	13			9	13	10	9	13	9	13			9	13		13	5

6. Définition des exigences du système.

Diagramme d'exigences :



7. Choix technologiques.

a. Proposition de solutions technologiques.

Pour les solutions techniques, nous avons fait le choix pour la centrale de données d'utiliser un Raspberry Pi. Nous avons utilisé le language SQL pour la base de données, le language c++ pour les programmes de la gestion des données et de l'affichage local et sur windows, le language Java pour le programme de l'affichage sur Android et les languages html et php pour la gestion du site web en ligne.

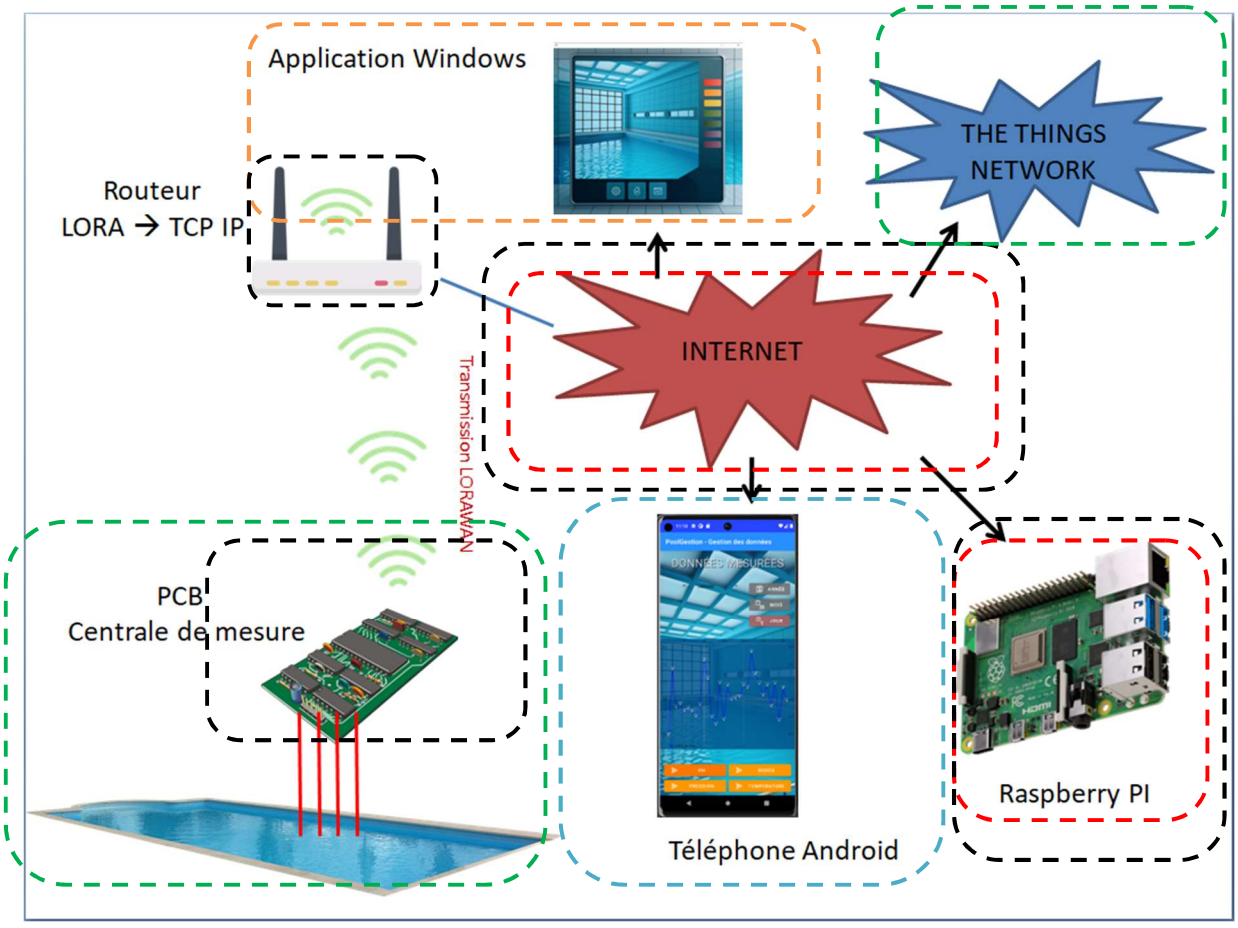
b. Choix technologique.

Dans cette partie, les solutions technologiques seront comparées et un choix justifié sera réalisé.

Dans la partie des eleves de IR, on choisit le raspberry car il nous était imposés. On a choisi nos différents languages informatiques dans nos programmes dû soit à la facilité d'utilisation ou la praticité de ces languges pour faire exactement ce dont on a besoin.

8. Définition de l'architecture du système

a. Situation de chaque partie personnelle dans l'ensemble du projet.



Encadrés en rouge, c'est le travail de Thomas POTON qui a crée un système de configuration embarqué sur la carte Rapberry Pi permettant depuis un site web de configurer la carte et d'accéder aux mesures de la base de données.

En noir, le travail de Nathan COSTES qui a mis en place la centrale de mesure, avec son serveur de base de données, sa base de données et son affichage.

Le travail a été aussi de lire les données envoyées par les élèves de EC afin de les transmettre dans la base de données sur le serveur. Il a aussi fallu mettre en place la machine virtuelle.

En bleu, le travail de Tommy HADDADOU qui a géré une interface Android afin d'afficher et de pouvoir lire les données avec un téléphone portable.

En orange, le travail de Florent FRADIER qui a crée une interface Windows pour lire et afficher les données avec des courbes.

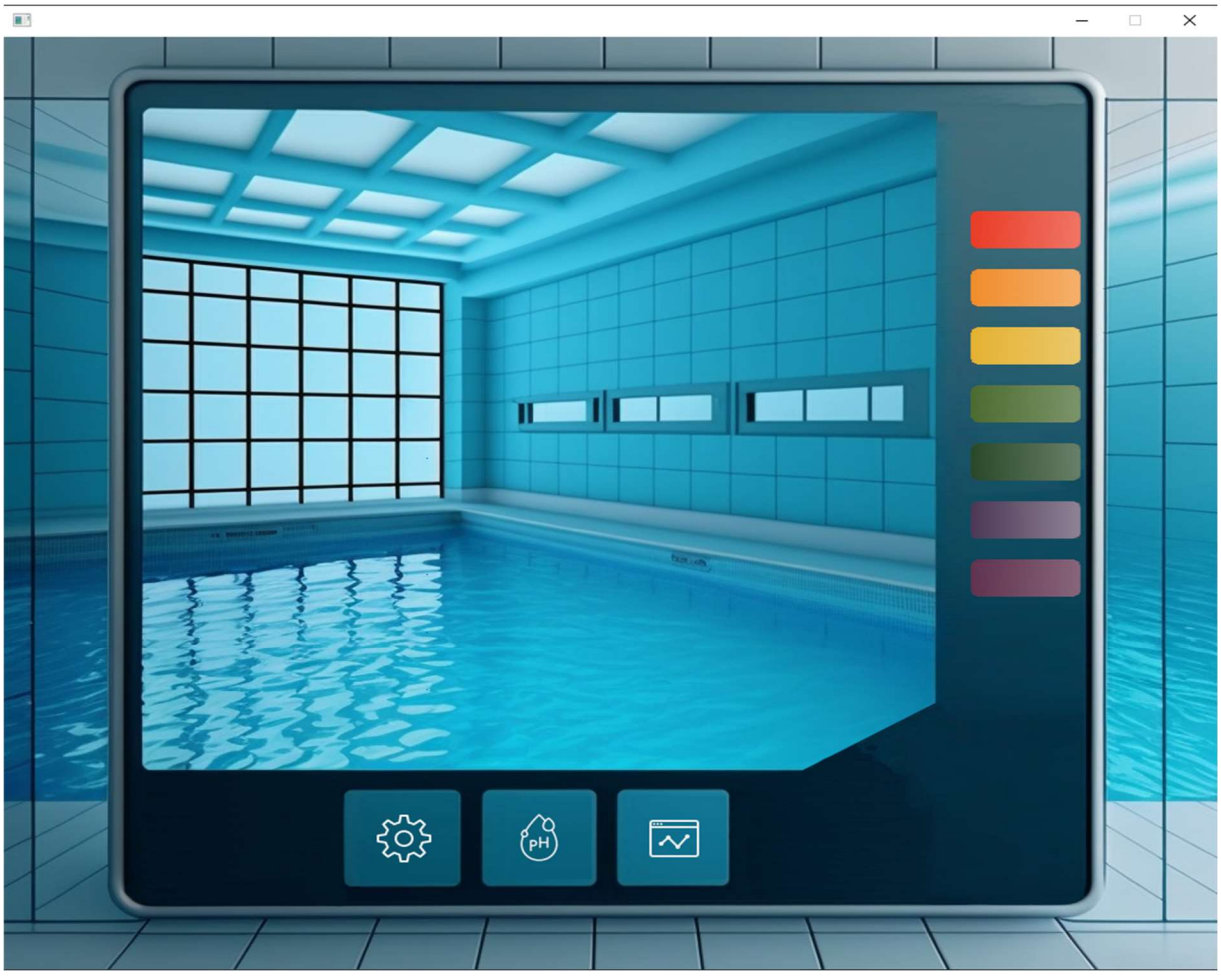
En vert, le travail de Christian NDINGA et Theo DUZELIER est de récupérer les mesures voulues (température de l'eau, pression de l'eau, REDOX et PH).Pour cela, ils ont du créer une carte car ils doivent traiter le signal reçue des sondes ou des capteurs. Ensuite, ils envoient les données sur TheThingsNetwork.

b. Modification éventuelle du cahier des charges en fonction des choix technologiques ou fonctionnels

Sur l'affichage du Raspberry (affichage local), il n'a pas pu être fait les courbes, uniquement les valeurs numériques.

RAPPORT DE PROJET INDIVIDUEL

Centrale de mesure autonome pour piscine par le biais d'un développement d'une application Windows



Sommaire

1. Dossier présentation du système technique

Ma partie consiste au développement d'une application pour Windows conçue pour configurer le système et afficher en temps réel les données mesurées par la centrale de mesure. Les informations seront présentées au sein d'une interface utilisateur graphique (IHM).

a. Cahier des charges de la partie étudiée par l'étudiant.

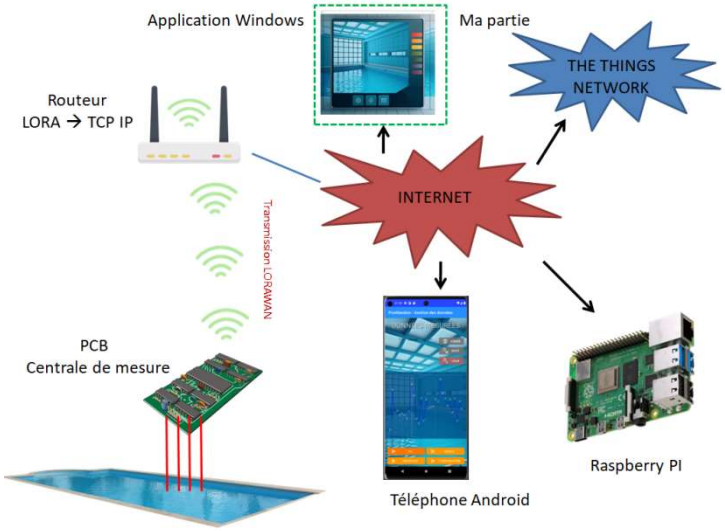
b. Diagramme d'exigences SysML de la partie traitée.

2. Dossier d'étude

Le développement est découpé en plusieurs parties, elles-mêmes contenant plusieurs Objectifs simples afin de réaliser les fonctionnalités demandées

Partie 1 :
- Création de la partie graphique
- Connexion a la base de donné

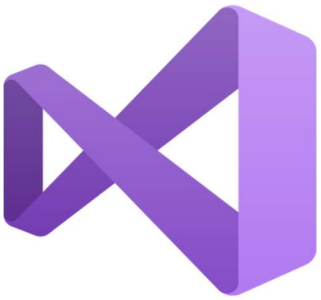
Une fois la communication réussis avec la base de données, je suis passé sur la partie traitement des donnés pour les afficher.



Partie 2 :
- Configuration du Calendrier pour traiter les données en fonction des jours sélectionnée

3. Présentation des logicielles et outils utilisés

Visual Studio est un environnement de développement intégré (IDE) créé par Microsoft. Visual Studio offre un ensemble d'outils de programmation, de débogage, de test et de gestion de projets
J'ai décidé de choisir cette IDE car je l'utilise personnellement je suis donc Habituer



Le langage utilisé pour le développement de cette partie est le C++ programmation orientée objet

Pour la partie graphique j'ai utilisé SDL qui est une bibliothèque open-source et multiplateforme qui offre un accès de bas niveau aux graphismes et pour interagir avec les périphériques d'entrée
De plus cette librairie est bien documentée

MySQL est une bibliothèque open-source qui permet de créer, gérer et interroger des bases de données relationnelles de manière efficace et sécurisée.

4. Diagramme de classes

Mon programme est constitué de 4 classes et de mon main

```
Calendrier anne2023(7, 2023, pRenderer);//Creation du calendrier
```

Pour créer un calendrier on doit juste préciser a partir de quand commence le premier jour de l'année dans la semaine et ensuite l'année pour lui permettre de s'adapter en fonction des années bissextile et SDL_Renderer pour la zone d'affichage

Lors de l'exécution on initialise la SDL , si elle ne charge pas correctement le logiciel s'arrête en renvoyant une erreur .

```
int sdlInitialisation()
{
    if (SDL_Init(SDL_INIT_VIDEO) < 0)
    {
        SDL_LogError(SDL_LOG_CATEGORY_APPLICATION, "[DEBUG] > %s", SDL_GetError());
        return EXIT_FAILURE;
    }

    if (SDL_CreateWindowAndRenderer(width, height, SDL_WINDOW_SHOWN, &pWindow, &pRenderer) < 0)
    {
        SDL_LogError(SDL_LOG_CATEGORY_APPLICATION, "[DEBUG] > %s", SDL_GetError());
        SDL_Quit();

        return EXIT_FAILURE;
    }
    TTF_Init();
    string path = "arial.ttf";
    font = TTF_OpenFont(path.c_str(), 36);
    if (!font) {
        std::cerr << "Error loading font: " << TTF_GetError() << std::endl;
        return 0;
    }

}
```

Ensuite loadTexture qui permet de charger l'affichage de mon application que j'ai créer grâce a Paint.net que l'on verra par la suite

```
loadTexture();
```

Quand le code s'exécute on arrive sur cette page de connexion on doit rentrer l'ip ,l'username , et le password j'ai ajouté une fonctionnalité de remember me et auto connect

Le code qui permet de lire le fichier texte lors de l'exécution pour remplir les case en fonction de ce que contient le fichier en question.

```
string dataLine;
int numberLine =0,stateRememberMe=0;
stringstream ss;
ifstream fileData;
fileData.open("bdd.txt");
while (getline(fileData, dataLine))
{
    if (numberLine == 0)
    {
        if (dataLine == "10") {
            stateRememberMe = 1;
            rememberMe = 1;
        }
        if (dataLine == "11") {
            stateRememberMe = 2;
            rememberMe = 1;
            autoConnect = 1;
        }
    }
    if (stateRememberMe ==1 || stateRememberMe ==
    {
        if (numberLine == 1)
            ip = dataLine;
        if (numberLine == 2)
            user = dataLine;
        if (numberLine == 3)
            pass = dataLine;
    }
    cout << dataLine << endl;
    numberLine++;
}
if (stateRememberMe==2)
{
    connectionBdd = 1;
    cout << server << endl;
    cout << username << endl;
    cout << password << endl;
}
```

Ici le code permet de modifier le fichier texte quand on modifie les paramètres de connexion :

```
    string saveFile = "bdd.txt";
    ofstream fileData;
    fileData.open(saveFile);
    if (rememberMe==1 && autoConnect==1)
        fileData << rememberMe << autoConnect<< "\n" << server << "\n" << username << "\n" << password; // on ajoute les identifiants dans le fichier texte
    if(rememberMe==1 &&autoConnect ==0)
        fileData << rememberMe << 0 << "\n" << server << "\n" << username << "\n" << password; // on ajoute les identifiants dans le fichier texte

    if (rememberMe == 0 && autoConnect == 0)
        fileData << "00" << "\n"; // on ajoute les identifiants dans le fichier texte

    fileData.close(); // fermeture du fichier
```

La fonction qui permet de se connecter au serveur de la base de données :

```
int bddConnection()
{
    int state = -1;
    cout << "Connexion SQL en cours..." << endl;
    try
    {
        driver = get_driver_instance();
        connection = driver->connect(server, username, password);
        cout << "Connexion succeed" << endl;
        connection->setSchema("piscine");
        state = 1;

    }
    catch (sql::SQLException e)
    {
        cout << "Connexion failed" << endl;
        state = 0;
    }
    return state;
}
```

Une fois connecter a la base de données la fonction loadData est exécuter j'aide décider pour des raisons d'optimisation de mon application de récupérer les données que une fois et ensuite les stockées dans la classe data

```
req = "SELECT * FROM mesure2 WHERE date LIKE '2023-";
req += date;
req += "%'";
```

Cette requête permet d'interroger la base de données en fonction de chaque mois il faut s'avoir que les données sont enregistrer toutes les 15 min une journée est composé de 4*24 = 96 data
La boucle while rempli chaque jour avec les 96 data et une fois remplie on passe au jour suivant et quand le mois est rempli la fonction loadData s'exécute au total 12 fois

```
if (stateConnection ==1)
{
    for (int i = 0; i < 12; i++)
    {
        loadData(&anne2023, date[i]);
    }
}
```

```
void loadData(Calendrier* p, string date)
{
    string req;

    req = "SELECT * FROM mesure2 WHERE date LIKE '2023-";
    req += date;
    req += "%'";
    prepStat = connection->prepareStatement(req);
    result = prepStat->executeQuery();

    vector<Data*> data = {};
    data.push_back(new Data[96]);
    int nombreDataJour = 0, jour = 0;

    while (result->next())
    {
        Data D(stoi(result->getString(5).substr(11, 2)), result->getDouble(1), result->getDouble(3), result->getDouble(2), result->getDouble(4));
        data[jour][nombreDataJour] = D;
        nombreDataJour++;
        if (nombreDataJour == 96)
        {
            p->getTabMois()[stoi(date) - 1].getTabJours()[jour].setData(data[jour]);
            nombreDataJour = 0;
            jour++;
            data.push_back(new Data[96]);
        }
    }
}
```

La partie qui affiche un graphique des données se présente comme ceci

On a le choix d'afficher la moyenne de chaque jours en fonction du mois choisit
On peut aussi choisir une plage de plusieurs jours (par exemple du 1 au 8)

Et pour finir la moyenne de chaque heure en fonction d'une journée

Quand on sélectionne une journée cette partie du code est exécuter ce qui permet De récupérer la moyenne du jour

```
anne2023.getTabMois()[numeroMois].getTabJours()[i].getMoyennePh(&value);
```

Le code :

```
void Jour::getMoyennePh(std::vector<float>* value)
{
    value->clear();
    float* moyenneDay[24] = {};
    float moyenne = 0;
    int compteur = 0;
    for (int i = 0; i < 24; i++)
    {
        for (int o = 0; o < 4; o++)
        {
            moyenne += m_tabData[compteur].getPh();
            compteur++;
        }
        moyenne = moyenne / 4;
        value->push_back(moyenne);
        moyenne = 0;
    }
}
```

Pour créer la courbe j'ai créer une fonction Graph

```
graph(typeGraph, 200, 580, rangeX, rangeY);//genere le graphique
```

Qui a plusieurs paramètres :

1)type graph un int qui correspond soit au Ph ,Redox ,Pression ,Température
2)Coordonnés X du graph
3) Coordonnés y du graph
4)rangeX = nombre de graduation sur l'axe X
5)rangeY = nombre de graduation sur l'axe Y

La composition de la fonction Graph

```c
int graph(int typeGraph, int x, int y, int rangeX, int rangeY)// FONCTION GRAPHIQUE
{
    SDL_SetRenderDrawColor(pRenderer, 255, 255, 255, 1);
    int offsetY = 0;
    int offsetX = 0;
    for (int i = 0; i < 3; i++) //trace l'abscisse  et l'ordonnée
    {
        SDL_RenderDrawLine(pRenderer, x + i, y, x + i, y - 430);
        SDL_RenderDrawLine(pRenderer, x, y + i, x + 520, y + i);
    }

    for (int i = 1; i <= rangeY; i++) //trace les trait ordonnée et graduation
    {
        int save = 430 / rangeY;
        offsetY = y - ((430 / rangeY) * i);
        SDL_RenderDrawLine(pRenderer, x, offsetY, x + 8, offsetY);
        if (typeGraph == 1){
            details = { x - 20,offsetY - 5, 13, 13 };
            SDL_RenderCopy(pRenderer, textureNumber[i - 1], NULL, &details);
        }
        if (typeGraph == 2){
            details = { x - 20,offsetY - 5, 13, 13 };
            SDL_RenderCopy(pRenderer, textureTemperature[i - 1], NULL, &details);
        }
        if (typeGraph == 3){
            details = { x - 20,offsetY - 5, 15, 15 };
            SDL_RenderCopy(pRenderer, textureRedox[i - 1], NULL, &details);
        }

        if (typeGraph == 4){
            details = { x - 20,offsetY - 5, 13, 13 };
            SDL_RenderCopy(pRenderer, texturePression[i - 1], NULL, &details);
        }
    }
    for (int i = 1; i < rangeX + 2; i++)//trace les trait abscisse et graduation
    {
        offsetX = x + ((520 / rangeX) * (i - 1));
        SDL_RenderDrawLine(pRenderer, offsetX, y - 5, offsetX, y);
        details = { offsetX - 5 ,y + 10, 13, 13 };
        SDL_RenderCopy(pRenderer, textureNumber[jourDebut + i - 1], NULL, &details);
    }
    offsetY = (430 / rangeY);
    offsetX = (520 / rangeX);
    for (int i = 0; i < rangeX; i++) //trace la courbe
    {
        if (typeGraph == 1) {
            SDL_RenderDrawLine(pRenderer, x + i * offsetX, y - offsetY * value[i], x + (i + 1) * offsetX, y - offsetY * value[i + 1]);
            Y1graphValue[i] = y - offsetY * value[i];
            Y2graphValue[i] = y - offsetY * value[i + 1];
        }
        if (typeGraph == 2) {
            SDL_RenderDrawLine(pRenderer, x + i * offsetX, y - offsetY * (value[i] + 20) / 5, x + (i + 1) * offsetX, y - offsetY * (value[i + 1] + 20) / 5);
            Y1graphValue[i] = y - offsetY * (value[i] + 20) / 5;
            Y2graphValue[i] = y - offsetY * (value[i + 1] + 20) / 5;
        }

        if (typeGraph == 3) {
            SDL_RenderDrawLine(pRenderer, x + i * offsetX, y - offsetY * value[i] / 50, x + (i + 1) * offsetX, y - offsetY * value[i + 1] / 50);
            Y1graphValue[i] = y - offsetY * value[i] / 50;
            Y2graphValue[i] = y - offsetY * value[i + 1] / 50;
        }

        if (typeGraph == 4) {
            SDL_RenderDrawLine(pRenderer, x + i * offsetX, y - offsetY * value[i] * 5, x + (i + 1) * offsetX, y - offsetY * value[i + 1] * 5);
            Y1graphValue[i] = y - offsetY * value[i] * 5;
            Y2graphValue[i] = y - offsetY * value[i + 1] * 5;
        }
        X1graphValue[i] = x + i * offsetX;
        X2graphValue[i] = x + (i + 1) * offsetX;
```

Les for() :

1)permet de tracer les abscisse et l'ordonnée avec une largeur de 3 pixels
2) trace la graduation sur les ordonnée
3) trace la graduation sur les abscisses
4) trace la courbe en fonction du type de graph

Le curseur qui nous donne les valeurs exactes

```c
Yoffset = Y1graphValue[i] - ((events.motion.x - X1graphValue[i]) * ((Y1graphValue[i] - Y2graphValue[i]) / (X2graphValue[i] - X1graphValue[i])));
Xoffset = events.motion.x;
```

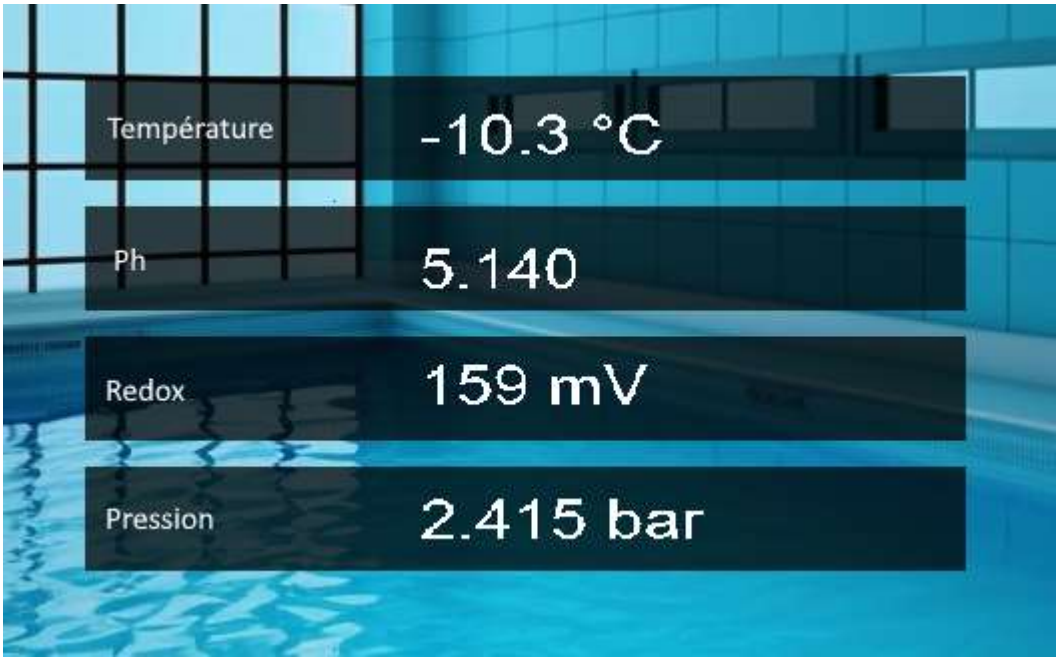
Yoffset calcule la coordonnée de Y pour placer le curseur en fonction de l'axe X

La fonction qui trace les points tillés en fonction des valeurs calculer juste au-dessus

```
void newCurseur(int x1, int y1) {
    SDL_SetRenderDrawColor(pRenderer, 255, 128, 0, 1);


    for (int i = x1; i < 720; i += 15) {
        SDL_RenderDrawLine(pRenderer, i, y1, i + 5, y1);
    }
    for (int i = x1; i > 200; i -= 15) {
        SDL_RenderDrawLine(pRenderer, i, y1, i - 5, y1);
    }
    for (int i = y1; i < 580; i += 15) {
        SDL_RenderDrawLine(pRenderer, x1, i, x1, i+5);
    }
    for (int i = y1; i > 150; i -= 15) {
        SDL_RenderDrawLine(pRenderer, x1, i, x1, i - 5);
    }
    SDL_SetRenderDrawColor(pRenderer, 255, 255, 255, 1);
}
```

Nous avons fini de voir les options du graphique nous allons maintenant voir la partie qui permet d'afficher en temps réel les dernières données qui ont était ajouter comme ci-dessous :



Cette fonction est situer dans un threads ce qui permet de faire une requête SQL que lorsqu'on appuie sur le bouton pour optimiser le programme , la requête :

```
string req = "SELECT* FROM mesure2 ORDER BY date DESC LIMIT 1";
```

Permet de récupérer la dernière donné de la base

Le while récupère tout ce qui est dans cette donnée sachant qu'elle est constituée ainsi.

	ph	temperature	redox	pression	date	ID
Modifier Copier Effacer	0.96	49.9166	383	2.53264	2023-01-01 00:00:00	1

Je place chacune des données dans une fonction qui lui correspond comme le Ph ,redox etc..

```
void dataTempReel()
{

    trHasToBeClear = 1;
    cout << "THREAD" << endl;
    int c = 0;
    string req = "SELECT* FROM mesure2 ORDER BY date DESC LIMIT 1";

    prepStat = connection->prepareStatement(req);
    result = prepStat->executeQuery();

    while (result->next())
    {
        ph = result->getString(1).substr(0, 5);
        redox =  result->getString(3).substr(0, 5);
        redox += " mV";
        temperature =  result->getString(2).substr(0, 5);
        temperature += " °C";
        pression =  result->getString(4).substr(0, 5);
        pression += " bar";
        cout << ph << " " << redox << " " << temperature << " " << pression << endl;
    }
    DirectData = true;
    cout << "FIN DU THREAD" << endl;
    trHasToBeClear = 2;

}
```

5. Analyse du coût

A ce jour nous avons passé environ 150 heures sur ce projet. Le smic brut étant actuellement à 11.52€, ce dernier reviendrait à 1728€ brut de travail. La carte nous a coûté environ 200€ ce qui revient à un coût de 1928€.

Afin de conclure, ce projet m'a permis de développer et révéler mon autonomie et ma rigueur. J'ai pu laisser place à ma créativité en prenant moi-même la décision de créer la partie graphique. Par cette totale liberté j'ai pu coder chaque option de mon application à savoir le calendrier et l'affichage des courbes, exemples qui m'ont permis de prendre la main sur la librairie SDL.

De plus, j'ai réussi à communiquer avec la base de données et ainsi les traiter. Or, les élèves étant chargés de la carte n'ont pas pu achever cette dernière dans son entièreté, ce qui entrainé le fait que les données dans la base ne proviennent pas de la carte.

Ce projet fut très enrichissant autant d'un point de vue personnel que professionnel. Il m'a notamment apporté des connaissances sur la façon d'entretenir une piscine ainsi que sur les différentes fonctionnalités de programmation.