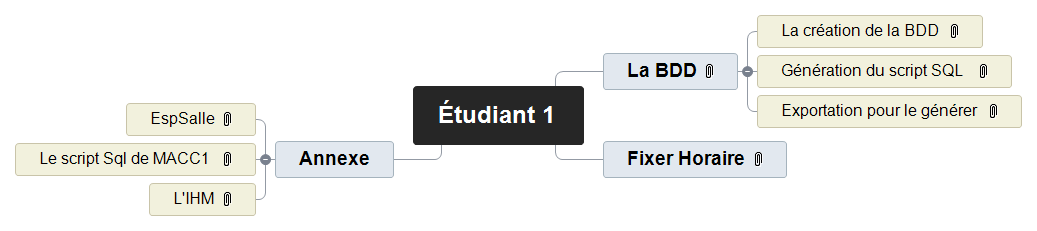
Étudiant 1



[La BDD 2](#_Toc4400034)

[La création de la BDD 4](#_Toc4400035)

[Génération du script SQL 8](#_Toc4400036)

[Exportation pour le générer 10](#_Toc4400037)

[Fixer Horaire 11](#_Toc4400038)

[Annexe 11](#_Toc4400039)

[EspSalle 11](#_Toc4400040)

[Le script Sql de MACC1 16](#_Toc4400041)

[L'IHM 19](#_Toc4400042)

# La BDD

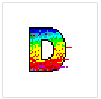
**L'idée de l'utilisation de BDD**

En se référant au cahier des charges, on y retrouve la mémorisation de données telles que les commandes de climatiseurs pour ainsi pouvoir les rejouer avec l'application de l'utilisateur. Mais, aussi il fallait stocker les identifiants des professeurs ainsi que les salles dans lesquels résident les microcontrôleurs avec leur adresse MAC. C'est ainsi qu'est venue donc la Base De Donnée qui sert a stocké toutes ces informations.

Pour pouvoir la mettre en production, il a fallu passer par la partie conceptuelle.

Un logiciel du nom de Win Design m’a permis de pouvoir créer des diagrammes de classe. Il s'agit de la version 9.0. Cet outil permet d’aussi de pouvoir générer différent autre type de diagramme que celui-ci. Tels que le modèle conceptuelle et logique. Et par la suite générer un code SQL.

Attention, ce logiciel a tout de même quelques difficultés au niveau des liaisons. Notamment, au niveau des diagrammes logiques, il y des liens direct entre objets qui n'est pas effectuer. Mais aussi, le modèle conceptuelle car en plus des liaisons manquantes il créer des associations par rapport aux attributs en communs.



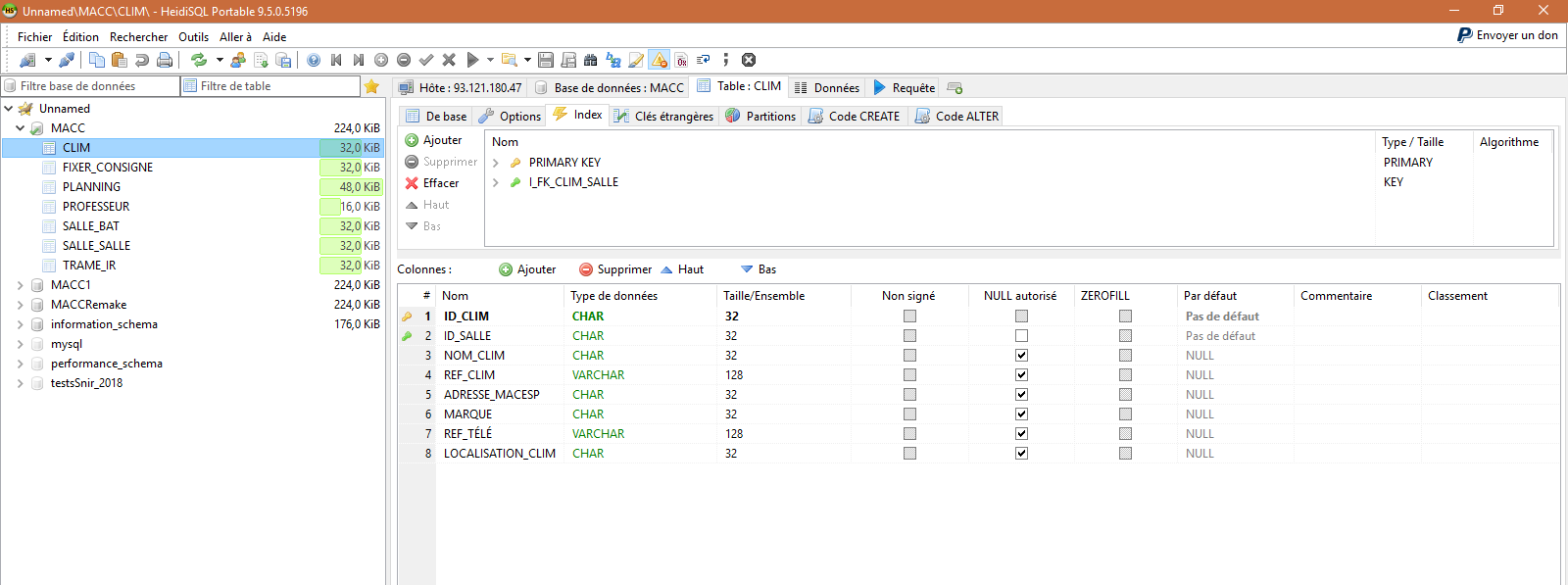
Icône de Win Design

Par la suite j'ai utilisé Heidi SQL, qui lui est un outil se connectant au serveur. Par ailleurs, pour ce projet la BDD a été créé sur un serveur Marida installé sur une raspberry. La particularité avec ce logiciel est que l'on peut créer des tables ainsi que gérer les données directement. Sans oublié que l'on peut nécessairement générer une BDD avec un script SQL.



Icône de Heidisql

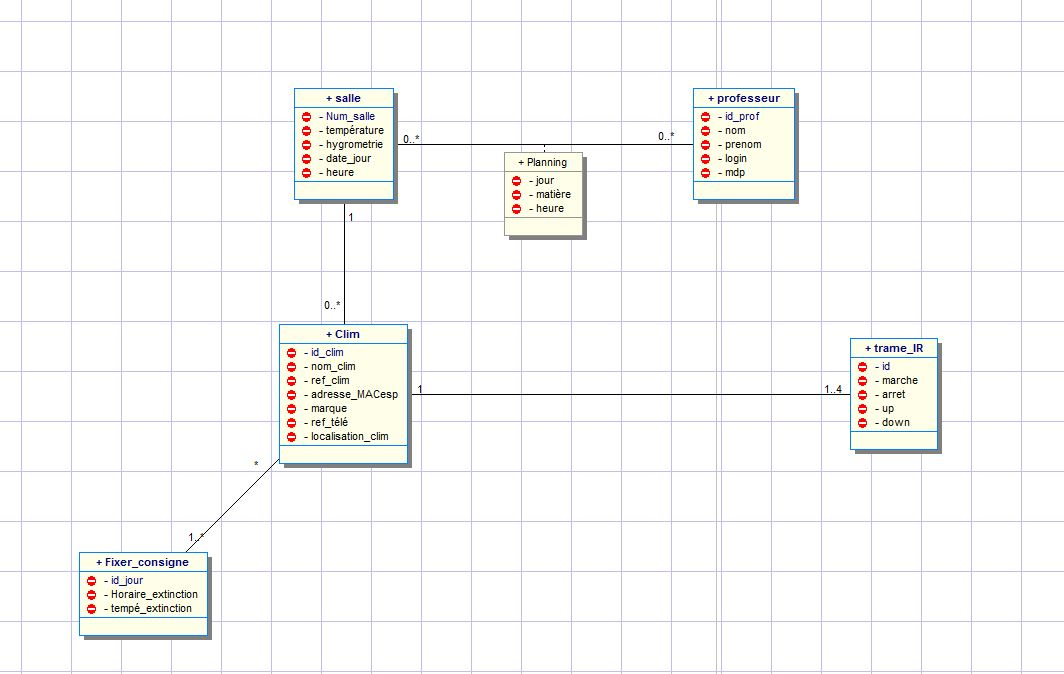
Voici un exemple d'utilisation de Heidisql avec la table CLIM:



## La création de la BDD

Voici donc le premier diagramme (qui correspond à la première synoptique):

Elle se nomme MACC\_BDD



Les significations des différentes tables:

- salle : Mémorise la température et l'humidité selon l'heure et la date.

-professeur : Tout ce qui concerne les données personnelles des professeurs.

-planning : Organise les professeurs pour qu'ils aient accès seulement aux climatiseurs de leur salle en fonction de leur emploi du temps.

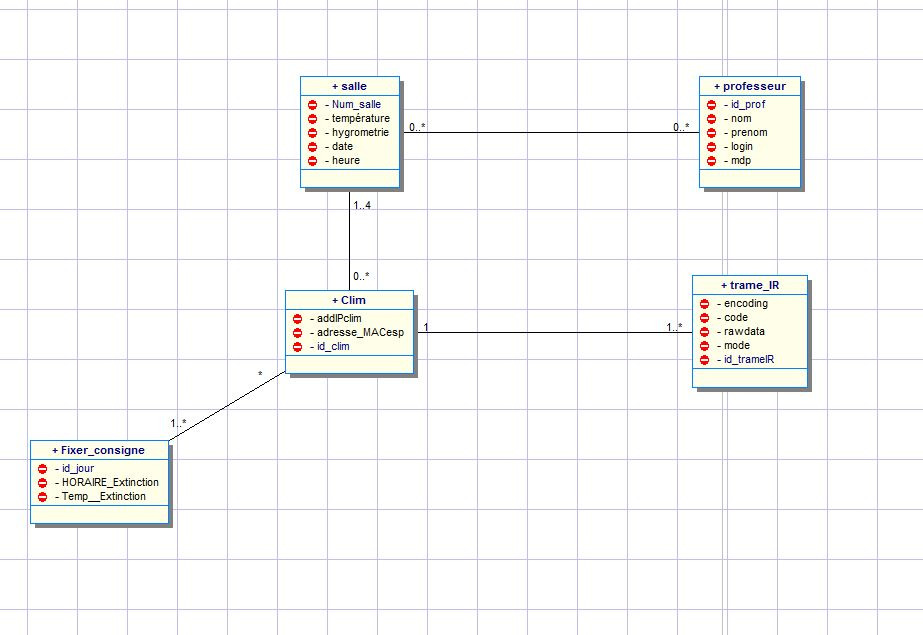
-Clim : Identifie les microcontrôleurs qui sont dans les salles.

-Trame IR : Elle sert à récupérer tous les données concernant les commandes des climatiseurs.

- Fixer consigne : Insère les directions pour savoir à quelles heures les climatiseurs s'éteindrons.

Puis par la suite, en vue du changement liés au projet, j'ai préféré crée une autre BDD. Mais aussi, parce qu’il y avait divers problèmes au niveau des données et des jointures.

On l'a nommé MACCRemake.

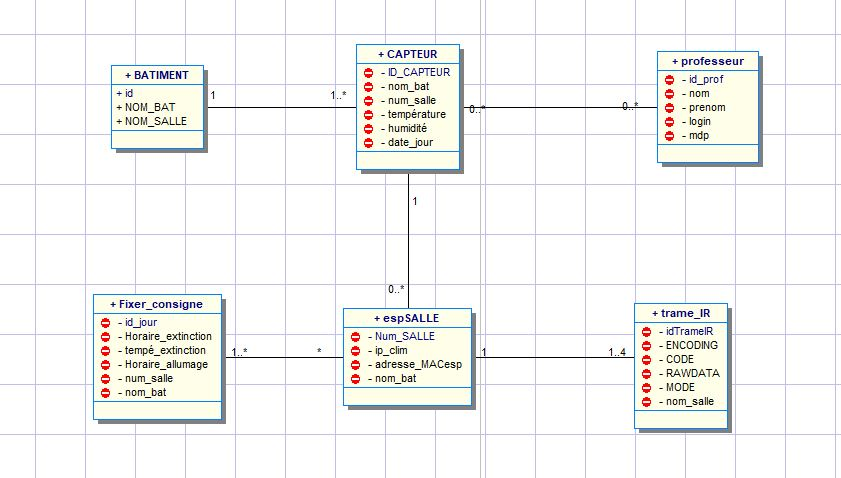


Le premier changement est la suppression de la table planning. Cela a été fait par rapport au fait qu'il n'était plus utile. En effet, on voulait plutôt retourner sur du basique en restant sur la sécurité par la connexion.

Aussi, au niveau de la table trame IR, les attributs ont changé parce qu’on a préféré stocker les données que les trames IR délivraient. Sauf pour "mode" qui lui sert a inséré la commande correspondante (On/Off, Up/Down).

Dans la table de Clim, on avait remarqué qu'il y avait beaucoup d'attributs qui n’étaient pas nécessaire. C'est pour cela que l'on a préféré garder justement seulement garder les adresses IP et MAC des microcontrôleurs (esp).

Enfin, la version final qui est MACC1 :

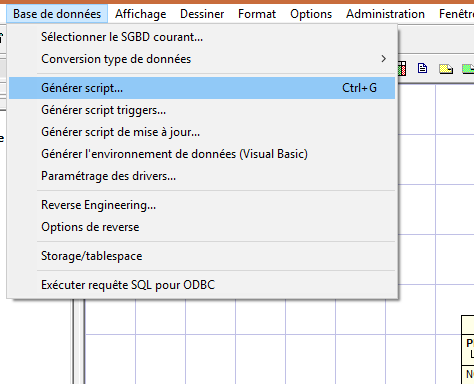


J'ai changé certaine classe telle que "Clim" qui est devenue "espSalle" ainsi que la table "salle" qui est devenue "Capteur". AU sujet des soucis de création de clé étrangère on a mis "nom\_bat" et "num\_salle" dans la table concernée. C'est pour cela que seulement dans la table "Batiment" les deux attributs sont en majuscules.

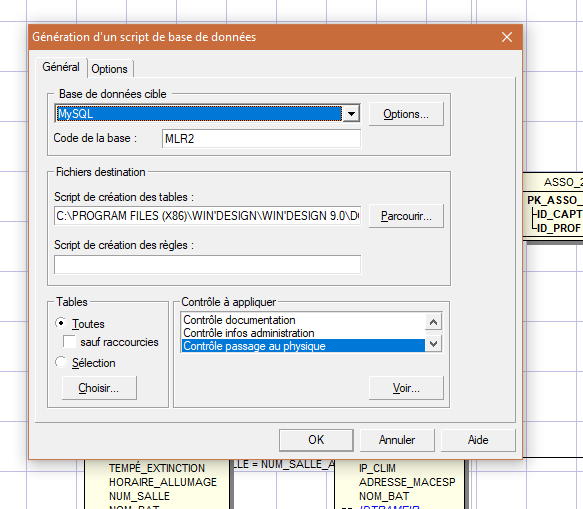
Viens ensuite, l'utilisation d’Heidi SQL.

## Génération du script SQL

Depuis Win design on peut générer un script SQL. Mais tout d'abord, il faut générer le modèle logique relationnelle pour ensuite générer le script comme indiquer-ci dessus:



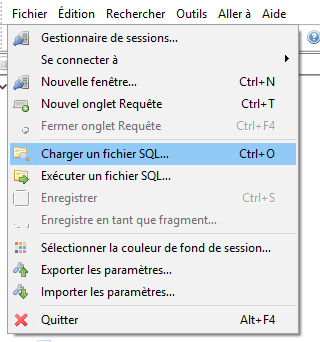
Après, sélectionner la base de donné ciblé ainsi que le fichier de destination:



Pour enfin avoir le script. (Voir annexe)

## Exportation pour le générer

Avec Heidi SQL, après avoir exporté le script via Win design il faut aller dans "Fichier" et sélectionner comme ci-dessous :



De cette manière, le logiciel récupère le script pour l'insérer dans la rubrique "Requête" et puis aller sur "Exécuter".

# Fixer Horaire

L'objectif de cette tâche est de pouvoir à travers l'IHM du superviseur pouvoir insérer des heures de démarrage et d'extinction de manière automatique.

# Annexe

## EspSalle

Création d'un code Arduino TestespSH pour pouvoir gérer les consignes de la BDD. Au final, le code arrive à se connecter et récupère la valeur horaire mais ne lance pas le minuteur. Le but est de pouvoir faire une comparaison entre l'heure de la requête et celui du minuteur.

Voici donc le code arduino avec lequel j'ai quelque test :

#include <WiFi.h>

#include <WiFiClient.h>

#include <WiFiServer.h>

#include <WiFiUdp.h>

#include <dummy.h>

#include <MySQL\_Connection.h>

#include <MySQL\_Cursor.h>

#include <MySQL\_Encrypt\_Sha1.h>

#include <MySQL\_Packet.h>

#include <Time.h>

#include <TimeLib.h>

const char ssid[] = "DESKTOP-GLPMSI"; // your network SSID (name)

const char pass[] = "97197170"; // your network password

IPAddress server\_addr(192, 168, 1, 8); // IP of the MySQL \*server\* here

char user[] = "pi"; // MySQL user login username

char password[] = "Simconolat"; // MySQL user login password

char EXEC\_SQL[128]; // Buffer de creation requete

char SELECT\_SQL[] = "SELECT horaire\_allumage FROM MACC.FIXER\_CONSIGNE WHERE ID\_JOUR=%d";

int cli=11;

char\* sel;

// NTP Servers:

static const char ntpServerName[] = "us.pool.ntp.org";

//static const char ntpServerName[] = "time.nist.gov";

//static const char ntpServerName[] = "time-a.timefreq.bldrdoc.gov";

//static const char ntpServerName[] = "time-b.timefreq.bldrdoc.gov";

//static const char ntpServerName[] = "time-c.timefreq.bldrdoc.gov";

const int timeZone = -5; // Central European Time

//const int timeZone = -5; // Eastern Standard Time (USA)

//const int timeZone = -4; // Eastern Daylight Time (USA)

//const int timeZone = -8; // Pacific Standard Time (USA)

//const int timeZone = -7; // Pacific Daylight Time (USA)

WiFiUDP Udp;

unsigned int localPort = 8888; // local port to listen for UDP packets

time\_t getNtpTime();

void digitalClockDisplay();

void printDigits(int digits);

void sendNTPpacket(IPAddress &address);

WiFiClient client;

MySQL\_Connection conn((Client \*)&client);

byte mac\_addr[] = { 0xDE, 0xAD, 0xBE, 0xEF, 0xFE, 0xED };

// Create an instance of the cursor passing in the connection

void setup() {

// put your setup code here, to run once:

Serial.begin(115200);

while (!Serial) ; // Needed for Leonardo only

delay(250);

Serial.println("TimeNTP Example");

Serial.print("Connecting to ");

Serial.println(ssid);

WiFi.begin(ssid, pass);

WiFi.setHostname("ESP\_Salle");

while (WiFi.status() != WL\_CONNECTED) {

delay(500);

Serial.print(".");

}

Serial.println("");

Serial.println("WiFi connected");

Serial.println("DB - Connecting...");

while (conn.connect(server\_addr, 3306, user, password) != true) {

delay(500);

Serial.print ( "." );

}

delay(1000);

sprintf(EXEC\_SQL, SELECT\_SQL, cli);

sel = SELECT(EXEC\_SQL); // exécute la fonction SELECT

conn.close();

Serial.println(sel);

}

time\_t prevDisplay = 0; // when the digital clock was displayed

void loop() {

// put your main code here, to run repeatedly:

if (timeStatus() != timeNotSet) {

if (now() != prevDisplay) { //update the display only if time has changed

prevDisplay = now();

digitalClockDisplay();

}

}

}

char\* SELECT(char\* recup)

{

// Serial.println("> Execution de SELECT with dynamically supplied parameter");

MySQL\_Cursor \*cur\_mem = new MySQL\_Cursor(&conn);

// preparer le code sql

// Execute the query

cur\_mem->execute(recup);

// Serial.println(EXEC\_SQL);

// Fetch the columns and print them

column\_names \*cols = cur\_mem->get\_columns();

// Serial.print("ColName:");

for (int f = 0; f < cols->num\_fields; f++) {

// Serial.print(cols->fields[f]->name);

if (f < cols->num\_fields - 1) {

// Serial.print(',');

}

}

// Serial.println();

// Read the rows and print them

row\_values \*row = NULL;

do {

row = cur\_mem->get\_next\_row();

if (row != NULL) {

for (int f = 0; f < cols->num\_fields; f++) {

if (f == 0) {

// Serial.print("\t");

}

char\* test = row->values[f];

//Serial.println(test);

return test;

if (f < cols->num\_fields - 1) {

// Serial.print(',');

}

}

// Serial.println();

}

}

while (row != NULL);

// Deleting the cursor also frees up memory used

delete cur\_mem;

}

void digitalClockDisplay()

{

// digital clock display of the time

Serial.print(hour());

printDigits(minute());

printDigits(second());

Serial.print(" ");

}

void printDigits(int digits)

{

// utility for digital clock display: prints preceding colon and leading 0

Serial.print(":");

if (digits < 10)

Serial.print('0');

Serial.print(digits);

}

/\*-------- NTP code ----------\*/

const int NTP\_PACKET\_SIZE = 48; // NTP time is in the first 48 bytes of message

byte packetBuffer[NTP\_PACKET\_SIZE]; //buffer to hold incoming & outgoing packets

time\_t getNtpTime()

{

IPAddress ntpServerIP; // NTP server's ip address

while (Udp.parsePacket() > 0) ; // discard any previously received packets

Serial.println("Transmit NTP Request");

// get a random server from the pool

WiFi.hostByName(ntpServerName, ntpServerIP);

Serial.print(ntpServerName);

Serial.print(": ");

Serial.println(ntpServerIP);

sendNTPpacket(ntpServerIP);

uint32\_t beginWait = millis();

while (millis() - beginWait < 1500) {

int size = Udp.parsePacket();

if (size >= NTP\_PACKET\_SIZE) {

Serial.println("Receive NTP Response");

Udp.read(packetBuffer, NTP\_PACKET\_SIZE); // read packet into the buffer

unsigned long secsSince1900;

// convert four bytes starting at location 40 to a long integer

secsSince1900 = (unsigned long)packetBuffer[40] << 24;

secsSince1900 |= (unsigned long)packetBuffer[41] << 16;

secsSince1900 |= (unsigned long)packetBuffer[42] << 8;

secsSince1900 |= (unsigned long)packetBuffer[43];

return secsSince1900 - 2208988800UL + timeZone \* SECS\_PER\_HOUR;

}

}

Serial.println("No NTP Response :-(");

return 0; // return 0 if unable to get the time

}

// send an NTP request to the time server at the given address

void sendNTPpacket(IPAddress &address)

{

// set all bytes in the buffer to 0

memset(packetBuffer, 0, NTP\_PACKET\_SIZE);

// Initialize values needed to form NTP request

// (see URL above for details on the packets)

packetBuffer[0] = 0b11100011; // LI, Version, Mode

packetBuffer[1] = 0; // Stratum, or type of clock

packetBuffer[2] = 6; // Polling Interval

packetBuffer[3] = 0xEC; // Peer Clock Precision

// 8 bytes of zero for Root Delay & Root Dispersion

packetBuffer[12] = 49;

packetBuffer[13] = 0x4E;

packetBuffer[14] = 49;

packetBuffer[15] = 52;

// all NTP fields have been given values, now

// you can send a packet requesting a timestamp:

Udp.beginPacket(address, 123); //NTP requests are to port 123

Udp.write(packetBuffer, NTP\_PACKET\_SIZE);

Udp.endPacket();

}

## Le script Sql de MACC1

DROP DATABASE IF EXISTS MACC1;

CREATE DATABASE IF NOT EXISTS MACC1;

USE MACC1;

# -----------------------------------------------------------------------------

# TABLE : PROFESSEUR

# -----------------------------------------------------------------------------

CREATE TABLE IF NOT EXISTS PROFESSEUR

(

ID\_PROF DB\_INTEGER(1) NOT NULL ,

NOM DB\_TEXT, 255 NULL ,

PRENOM DB\_TEXT, 255 NULL ,

LOGIN DB\_TEXT, 255 NULL ,

MDP DB\_TEXT, 255 NULL

, PRIMARY KEY (ID\_PROF)

)

comment = "";

# -----------------------------------------------------------------------------

# TABLE : TRAME\_IR

# -----------------------------------------------------------------------------

CREATE TABLE IF NOT EXISTS TRAME\_IR

(

IDTRAMEIR DB\_INTEGER(1) NOT NULL ,

ID\_CLIM DB\_TEXT, 32 NOT NULL ,

ENCODING DB\_TEXT, 32 NULL ,

CODE DB\_TEXT, 32 NULL ,

RAWDATA DB\_TEXT, 32 NULL ,

MODE DB\_TEXT, 32 NULL ,

NOM\_SALLE DB\_TEXT, 32 NULL

, PRIMARY KEY (IDTRAMEIR)

)

comment = "";

# -----------------------------------------------------------------------------

# TABLE : BATIMENT

# -----------------------------------------------------------------------------

CREATE TABLE IF NOT EXISTS BATIMENT

(

ID DB\_TEXT, 32 NOT NULL ,

NOM\_BAT DB\_TEXT, 32 NULL ,

NOM\_SALLE DB\_TEXT, 32 NULL

, PRIMARY KEY (ID)

)

comment = "";

# -----------------------------------------------------------------------------

# TABLE : ESPSALLE

# -----------------------------------------------------------------------------

CREATE TABLE IF NOT EXISTS ESPSALLE

(

ID\_CLIM DB\_TEXT, 32 NOT NULL ,

ID\_SALLE DB\_TEXT, 32 NOT NULL ,

NOM\_CLIM DB\_TEXT, 32 NULL ,

ADRESSE\_MACESP DB\_TEXT, 32 NULL ,

MARQUE DB\_TEXT, 32 NULL ,

LOCALISATION\_CLIM DB\_TEXT, 32 NULL

, PRIMARY KEY (ID\_CLIM)

)

comment = "";

# -----------------------------------------------------------------------------

# TABLE : SALLE

# -----------------------------------------------------------------------------

CREATE TABLE IF NOT EXISTS SALLE

(

ID\_SALLE DB\_TEXT, 32 NOT NULL ,

ID DB\_TEXT, 32 NOT NULL ,

TEMPÉRATURE DB\_DOUBLE(10,2) NULL ,

HUMIDITÉ DB\_DOUBLE(10,2) NULL ,

DATE\_JOUR DB\_DATE(8) NULL

, PRIMARY KEY (ID\_SALLE)

)

comment = "";

# -----------------------------------------------------------------------------

# TABLE : FIXER\_CONSIGNE

# -----------------------------------------------------------------------------

CREATE TABLE IF NOT EXISTS FIXER\_CONSIGNE

(

ID\_JOUR DB\_INTEGER(1) NOT NULL ,

ID\_CLIM DB\_TEXT, 32 NULL ,

HORAIRE\_EXTINCTION DB\_DATE(4) NULL ,

TEMPÉ\_EXTINCTION DB\_DOUBLE(10,2) NULL ,

HORAIRE\_ALLUMAGE DB\_DATE(4) NULL

, PRIMARY KEY (ID\_JOUR)

)

comment = "";

# -----------------------------------------------------------------------------

# TABLE : ASSO\_2

# -----------------------------------------------------------------------------

CREATE TABLE IF NOT EXISTS ASSO\_2

(

ID\_SALLE DB\_TEXT, 32 NOT NULL ,

ID\_PROF DB\_INTEGER(1) NOT NULL

, PRIMARY KEY (ID\_SALLE,ID\_PROF)

)

comment = "";

# -----------------------------------------------------------------------------

# CREATION DES REFERENCES DE TABLE

# -----------------------------------------------------------------------------

ALTER TABLE TRAME\_IR

ADD FOREIGN KEY FK\_TRAME\_IR\_ESPSALLE (ID\_CLIM)

REFERENCES ESPSALLE (ID\_CLIM) ;

ALTER TABLE ESPSALLE

ADD FOREIGN KEY FK\_ESPSALLE\_SALLE (ID\_SALLE)

REFERENCES SALLE (ID\_SALLE) ;

ALTER TABLE SALLE

ADD FOREIGN KEY FK\_SALLE\_BATIMENT (ID)

REFERENCES BATIMENT (ID) ;

ALTER TABLE FIXER\_CONSIGNE

ADD FOREIGN KEY FK\_FIXER\_CONSIGNE\_ESPSALLE (ID\_CLIM)

REFERENCES ESPSALLE (ID\_CLIM) ;

ALTER TABLE ASSO\_2

ADD FOREIGN KEY FK\_ASSO\_2\_SALLE (ID\_SALLE)

REFERENCES SALLE (ID\_SALLE) ;

ALTER TABLE ASSO\_2

ADD FOREIGN KEY FK\_ASSO\_2\_PROFESSEUR (ID\_PROF)

REFERENCES PROFESSEUR (ID\_PROF) ;

## L'IHM

Conception de l'IHM avec l'étudiant 2 pour le superviseur pour fixé l'horaire et la température.

Il a été codé en PHP, HTML et le CSS

Il y a la page d'index.php :

<html>

<head>

<title> Identification supervision </title>

<meta charset="utf-8">

<!-- importer le fichier de style -->

<link rel="stylesheet" href="CSS.css" type="text/css" />

</head>

<body>

<center><h1>Superviseur</h1></center>

<div id="container">

<!-- zone de connexion -->

<form action="verification.php" method="POST">

<center> <h1>Connexion</h1></center>

<label><b>Nom d'utilisateur</b></label>

<input type="text" placeholder="Entrer le nom d'utilisateur" name="username" required>

<label><b>Mot de passe</b></label>

<input type="password" placeholder="Entrer le mot de passe" name="password" required>

<input type="submit" id='submit' value='LOGIN' >

<?php

if(isset($\_GET['erreur'])){

$err = $\_GET['erreur'];

if($err==1 || $err==2)

echo "<p style='color:red'>Utilisateur ou mot de passe incorrect</p>";

}

?>

</form>

</div>

</body>

</html>

Puis, il a celui de vérification.php. Il vérifie la connexion avec l'identifiant stocké dans la BDD.

<?php

session\_start();

if(isset($\_POST['username']) && isset($\_POST['password']))

{

// connexion à la base de données

$db\_user = 'pi';

$db\_password = 'Simconolat';

$db\_name = 'MACC';

// $db\_host = '192.168.13.91';

$db\_host = '93.121.180.47';

$db = mysqli\_connect($db\_host, $db\_user, $db\_password,$db\_name)

or die('could not connect to database');

// on applique les deux fonctions mysqli\_real\_escape\_string et htmlspecialchars

// pour éliminer toute attaque de type injection SQL et XSS

$username = mysqli\_real\_escape\_string($db,htmlspecialchars($\_POST['username']));

$password = mysqli\_real\_escape\_string($db,htmlspecialchars($\_POST['password']));

if($username !== "" && $password !== "")

{

$requete = "SELECT count(\*) FROM PROFESSEUR where

LOGIN = '".$username."' and MDP = '".$password."' ";

$exec\_requete = mysqli\_query($db,$requete);

$reponse = mysqli\_fetch\_array($exec\_requete);

$count = $reponse['count(\*)'];

if($count!=0) // nom d'utilisateur et mot de passe correctes

{

$\_SESSION['username'] = $username;

header('Location: formulaire.php');

}

else

{

header('Location: index.php?erreur=1'); // utilisateur ou mot de passe incorrect

}

}

else

{

header('Location: index.php?erreur=2'); // utilisateur ou mot de passe vide

}

}

else

{

header('Location: index.php');

}

mysqli\_close($db); // fermer la connexion

?>

Ensuite, le get.php sert a pouvoir donc insérer les consignes d'horaire et de température dans la BDD:

<?php

//on recupère les valeurs du formulaire

$allumage=$\_POST['allumage'];

$extinction=$\_POST['extinction'];

$temp=$\_POST['temp'];

$nom\_salle=$\_POST['nom\_salle'];

// connexion à la base de données

$db\_user = 'pi';

$db\_password = 'Simconolat';

$db\_name = 'MACC';

// $db\_host = '192.168.13.91';

$db\_host = '93.121.180.47';

$db = mysqli\_connect($db\_host, $db\_user, $db\_password,$db\_name)

or die("Connection failed: " . mysqli\_connect\_error());

$sql = "INSERT INTO FIXER\_CONSIGNE (`HORAIRE\_ALLUMAGE`, `HORAIRE\_EXTINC`, `TEMP\_FIXE`, `SALLE`)

VALUES ('".$allumage."','".$extinction."','".$temp."','".$nom\_salle."')";

if (mysqli\_query($db, $sql)) {

echo "New record created successfully";

} else {

echo "Error: " . $sql . "<br>" . mysqli\_error($db);

}

header('Refresh:0;url=fin.html');

mysqli\_close($db);

?>

Celui du getuser.php sert a pouvoir trouver la salle sur laquelle posé les consignes.

<!DOCTYPE html>

<html>

<head>

<style>

table {

width: 100%;

border-collapse: collapse;

}

table, td, th {

border: 1px solid black;

padding: 5px;

}

th {text-align: left;}

</style>

</head>

<body>

<?php

//$con = mysqli\_connect('192.168.13.91','pi','Simconolat','MACC');

$con = mysqli\_connect('93.121.180.47','pi','Simconolat','MACC');

if (!$con) {

die('Could not connect: ' . mysqli\_error($con));

}

mysqli\_select\_db($con,"");

$sql="SELECT \* FROM SALLE\_SALLE ";

$result = mysqli\_query($con,$sql);

echo "

</option>

<table>

<tr>

<th>NOM\_SALLE</th>

</tr>";

while($row = mysqli\_fetch\_array($result)) {

echo "<td>" . $row['NOM\_SALLE'] . "</td>";

echo "</tr>";

}

echo "</table>";

mysqli\_close($con);

?>

</body>

</html>