CODE PARC AQUATIQUE

Code Attraction

CLecteur.h

```
#ifndef CLECTEURRFID_H
#define CLECTEURRFID_H
#include "CMFRC522.h"
#include <pigpio.h>


class CLecteurRFID
{
    public:
        CLecteurRFID();
        ~CLecteurRFID();
        void fermerportspi();
// retourne numéro du badge
        long long int lirebadge();

    private:
        CMFRC522 mfrc522;

};

#endif
```

CLecteur.cpp

```
#include "CLecteurRFID.h"
#include <iostream>
#include <sstream>
#include <math.h>
#include "pigpio.h"
#include <unistd.h>
#include <wiringPi.h>
CLecteurRFID::CLecteurRFID()
{

    //constructeur
}

CLecteurRFID::~CLecteurRFID()
{
    //destructeur
}

void CLecteurRFID::fermerportspi()
{
    gpioTerminate();
}

long long int CLecteurRFID::lirebadge()
{
         long long int decimalValue=0;
            int rep=1;
                // Initialiation PCD
                mfrc522.PCD_Init();
                while(rep==1) {
                    if (mfrc522.isNewCardPresent()==0) { // Detection d'un bracelet
                        delay(5000);
                        if(mfrc522.PICC_ReadCardSerial());
                        uint8_t nuidPICC[5];
                  //récuperation des octets du badge de la trame
                        for (int i = 11; i < 16; ++i)
                        {
                            nuidPICC[i - 11] = mfrc522.uid.uidByte[i];
                        }

                        printf("\n");
                        //conversion du numéro de série vers decimal
                        for (int i = 0; i < 4; ++i) {
                        decimalValue += nuidPICC[i] << (8 * i);
                        }
                        return decimalValue;
                        }
                }

}
```

CBaseDeDonnees.h

```
#ifndef CBASE_DE_DONNEES_H
#define BASE_DE_DONNEES_H

#include <mysql/mysql.h>
#include <string>

class CBaseDeDonnees {
private:
    MYSQL mysql;
//*maBase est utiliser pour interagir avec la base de donnees lors de l'execution des requetes
    MYSQL *maBase;
// utiliser pour stocker le résultat des requêtes exécuté sur la base de données
    MYSQL_RES *res;

public:
    CBaseDeDonnees();
    ~CBaseDeDonnees();
    bool connexion(char *server,char *user,char *password, char *database);
    bool deconnexion();
    bool envoyerRequete(char *requete);
    MYSQL_RES *GetResponse();
};

#endif
```

CBaseDeDonnees.cpp

```
#include "CBaseDeDonnees.h"
#include <iostream>
#include "Requete.h"

CBaseDeDonnees::CBaseDeDonnees() {}

CBaseDeDonnees::~CBaseDeDonnees() {}

bool CBaseDeDonnees::connexion(char *server,char *user,char *password, char *database) {
    mysql_init(&mysql);
    maBase=mysql_real_connect(&mysql, server, user, password, database, 0, 0, 0);
    if (maBase==NULL){
        return false;
        }
    return true;
}


bool CBaseDeDonnees::deconnexion() {
    mysql_close(maBase);
    return true;
}

bool CBaseDeDonnees::envoyerRequete(char *requete) {

  int query_state;
    query_state=mysql_query(&mysql, requete);
    if (query_state!=0){
    std::cout << "Erreur lors de l'envoi de la requete : " << mysql_error(&mysql) <<
std::endl;
        return false;
        }
    res=mysql_store_result(maBase);
    return true;
}

MYSQL_RES *CBaseDeDonnees::GetResponse()
{
    return res;
}
```

CRequete.h

```
#ifndef CREQUETE_H
#define CREQUETE_H

#include <string>
#include <vector>
#include "CBaseDeDonnees.h"

class CRequete {
private:
// m_bdd” permet à la classe CRequete d'accéder et d'utiliser les fonctionnalités de la classe CBaseDeDonnees
    CBaseDeDonnees* m_bdd;
public:
    CRequete(CBaseDeDonnees* bdd);
    ~CRequete();
    bool verifierExistenceId(long long int numerobadge);
    int recupererTaille(long long int numerobadge);
    MYSQL_RES *REPONSE();
    bool ajouterAccesRefuser(long long int numerobadge);
    bool ajouterHistorique(long long int numerobadge);
};

#endif
```

CRequete.cpp

```
#include "CRequete.h"
#include "CBaseDeDonnees.h"
#include <iostream>
#include <chrono>

CRequete::CRequete(BaseDeDonnees* bdd) {
// initialise l'attribut pour effectuer des operations de requetes
    m_bdd = bdd;
}

CRequete::~CRequete() {}

bool CRequete::verifierExistenceId(long long int numerobadge) {
    char req[500];
sprintf(req, "SELECT ID FROM visiteurs WHERE ID='%lld'", numerobadge);
   printf("la requete est %s",req);
//  le numero du bracelet est valide retourne true si invalide retourne false
   if(m_bdd->envoyerRequete(req)){
   MYSQL_RES* resultat = REPONSE();
       if(resultat != NULL){
          MYSQL_ROW row = mysql_fetch_row(resultat);
          mysql_free_result(resultat);
          if(row != NULL){
             return true;
           }
       }
   }
   return false;
}

int CRequete::recupererTaille(long long int numerobadge) {
    char req[500];
    sprintf(req, "SELECT `taille` FROM `visiteurs` WHERE ID='%lld'", numerobadge);
    printf("la requete est %s",req);

    if (!m_bdd->envoyerRequete(req)) {
        printf("Erreur");
        return -1; // retourner une valeur d'erreur
    }

    MYSQL_RES *resultat = m_bdd->GetResponse();
    MYSQL_ROW row;
    int taille = -1;
    while ((row = mysql_fetch_row(resultat))) {
        taille = std::stoi(row[0]); // convertir la chaîne en entier

    }

    return taille;
}

bool CRequete::ajouterAccesRefuser(long long int numerobadge) {
    char req[500];
    sprintf(req, "UPDATE visiteurs SET NombreRefus = NombreRefus + 1 WHERE ID='%lld'",
numerobadge);
    printf("la requete est %s", req);
```

```
    return m_bdd->envoyerRequete(req);
}

bool CRequete::ajouterHistorique(long long int numerobadge) {
    char req[500];
     std::string attraction = "Toutatis";
    sprintf(req, "INSERT INTO refus(ID, HeureRefus, NomAttraction) VALUES ('%lld', now(),
'%s')", numerobadge, attraction.c_str());
    printf("la requete est %s", req);
    return m_bdd->envoyerRequete(req);
}


MYSQL_RES *CRequete::REPONSE() {
    return m_bdd->GetResponse(); // permet d'effectuer l'envoi de la requête SQL à la base de
données et de gérer le resultat de cette opération
}
```

CLed.h

```
#ifndef CLED_H
#define CLED_H

class CLed {
private:
//pin de la led GPIO
    int pin_;

public:
    CLed(int pin);
    void Vert();
    void Rouge();
};

#endif
```

CLed.cpp

```
#include "CLed.h"
#include <wiringPi.h>

CLed::CLed(int pin) {
// On assigne la valeur du pin à l'objet
    pin_ = pin;
    pinMode(pin_, OUTPUT);
}

void CLed::Vert() {
    digitalWrite(pin_, HIGH);
}

void CLed::Rouge() {
    digitalWrite(pin_, LOW);
}
```

main.cpp

```
#include <iostream>
#include <unistd.h>
#include "CLecteurRFID.h"
#include <wiringPi.h>
#include "CBaseDeDonnees.h"
#include "CRequete.h"
#include "CLed.h"
using namespace std;

void delay(int ms){
#ifdef WIN32
  Sleep(ms);
#else
  usleep(ms*1000);
#endif
}

int main(int argc, char *argv[]) {

CBaseDeDonnees bdd;
 if (!bdd.connexion("172.29.21.52", "AdminP", "Aparc", "parc_aquatique")) {
        std::cerr << "Erreur de connexion Ã  la base de donnÃ©es." << std::endl;
        return 1;
    }else{

           //initialisation du gpio
            if (gpioInitialise() < 0)
            {
                std::cerr << "Port SPI non initialisÃ©." << std::endl;

            }
            else
            {
         // declaration de l'objet du lecteur
              CLecteurRFID monlecteur;
             int retourhaut =1;
                // boucle infinie
              while(retourhaut != 0) {
                      // lecture du badge avec la methode lirebadge
                         long long int decimalValue = monlecteur.lirebadge();
                           if (decimalValue != 0)
                           {
                      printf("Numero de badge converti en base 10: %lli\n", decimalValue);
       // declaration de la variable CRequete avec en parametre adresse de l'objet bdd
                                CRequete req(&bdd);
                                bool existe = req.verifierExistenceId(decimalValue);
                     std::cout << "La valeur du boolÃ©en est : " << existe << std::endl;
                            //verification de la validitÃ© du bracelets
                               if(existe == true)
                                {
                                   std::cout << "L'ID " << decimalValue << " existe dans
                                  la base de donnÃ©es." << std::endl;
                              //recuperation de la taille
                                        int taille = req.recupererTaille(decimalValue);
                                        std::cout << "Taille : " << taille << std::endl;
                             // j'ai fixer la taille a 175 cm pour l'acces a l'attraction
                                        if(taille >= 175)
```

```
                                {
                        // si la taille est correct,on affiche la led en vert

                                    wiringPiSetup();
                                    CLed ma_led(29);
                                    ma_led.Vert();
                                    delay(600);
                                    ma_led.Rouge();

                                }else{
                                   req.ajouterAccesRefuser(decimalValue);
                                   req.ajouterHistorique(decimalValue);

                                    break;
                                }

                            }
                            else
                            {
                              std::cerr << "L'ID " << decimalValue << " n'existe
                              pas dans la base de donnees." << std::endl;
                            }
                        }
                         delay(1000);
                         retourhaut =1;
                    }
                monlecteur.fermerportspi();
        }
    }
}
```

Code site internet

Insciption.php

```
<!DOCTYPE html>

<html>

  <head>
    <link rel="stylesheet" type="text/css" href="CSSV2.css">
    <meta charset="UTF-8">
    <title>O Gliss park</title>
  </head>
  <body>
    <div class="image" style="background-image: url('FondV2.jpg');">
      <header>
        <div class="titre"><h1>Bienvenue sur votre espace personnel</h1></div>
        <img src="OGliss.png" alt="Logo" style="float: left; margin-right: 1em; width:
227px; height: 90px;">
      </header>
      <nav>
        <ul>
          <li><a href="#about">À propos</a></li>
          <li><a href="#informations">Informations</a></li>
        </ul>
      </nav>
      <main>
        <section id="about">
          <h2>À propos</h2>
          <p>Bienvenue dans notre parc aquatique. Veuillez s'il vous plaît renseigner vos
données avec le plus de précision possible. Merci.</p>
        </section>
        <section id="informations">
          <h2>Informations</h2>
          <form action="traitement.php" method="post">
            <label for="nom">Nom :</label>
            <input type="text" id="nom" name="nom"><br>

            <label for="prenom">Prénom :</label>
            <input type="text" id="prenom" name="prenom"><br>


            <label for="motdepasse">Mot de passe :</label>
            <input type="password" id="motdepasse" name="motdepasse"><br>


            <label for="taille">Taille :</label>
            <input type="text" id="taille" name="taille"><br>


            <label for="ID">ID :</label>
            <input type="text" id="ID" name="ID"><br>
            <button type="submit" name="submit">Envoyer</button>
            </form>
        </section>
      </main>
      <footer>
        <p>Respect des normes RGPD : Ce site web respecte les normes de protection de la
vie privée et de la protection des données personnelles conformément au RGPD.</p>
      </footer>
    </div>
  </body>
</html>
```

Traitement.php

```php
<?php


$host = 'localhost';
$username = 'root';
$password = '';
$database = 'parc_aquatique';


// Connexion a la base de donnees
$conn = mysqli_connect($host, $username, $password, $database);


// Verifier la connexion
if (!$conn) {
    die("La connexion a la base de donnees a echoue : " . mysqli_connect_error());
}




// Recuperer les donnees saisies
$nom = $_POST['nom'];
$prenom = $_POST['prenom'];
$motdepasse = $_POST['motdepasse'];
$taille = $_POST['taille'];
$ID = $_POST['ID'];


// Preparer la requete
$stmt = mysqli_prepare($conn, "INSERT INTO visiteurs (nom, prenoms, mot_de_passe, taille,
ID) VALUES (?, ?, ?, ?, ?)");


// Lier les parametres a la requete preparee
mysqli_stmt_bind_param($stmt, "ssssi", $nom, $prenom, $motdepasse, $taille, $ID);


// Executer la requete
if (mysqli_stmt_execute($stmt)) {
    echo "Les donnees ont ete inserees avec succes.";
} else {
    echo "Erreur : " . mysqli_error($conn);
}




// Fermer la connexion a la base de donnees
mysqli_close($conn);


?>
```

ConnexionWeb.php

```
<!DOCTYPE html>
<html>
<head>
    <title>Login</title>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1">
    <link rel="stylesheet" type="text/css" href="css/main.css">
</head>
<body>
    <form method="post" action="connexion.php">


        <div class="image" style="background-image: url('images/FondV2.jpg');">
            <div class="wrap-login100 p-t-30 p-b-50">
                <span class="titre">
                    Connexion
                </span>
                <div class="zoneBlanche">


                    <div class="zone" data-validate="nom">
                    <input class="text" type="text" id="nom" name="nom" placeholder="Nom">
                    </div>


                    <div class="zone" data-validate="password">
                    <input class="text" type="password" id="password" name="password"
placeholder="Mot de passe">
                    </div>

                    <div class="confirme">
                        <input class="couleurbtn" type="submit" name="submit" value="Se
connecter"></input>
                    </div>
                </div>
            </div>
        </div>
    </form>
</body>
</html>
```

Connexion.php

```php
<?php


// Debut de la session
session_start();


$host = 'localhost';
$username = 'root';
$password = '';
$database = 'parc_aquatique';


// Connexion a la base de donnees
$conn = mysqli_connect($host, $username, $password, $database);


// Recuperer les informations du formulaire
$username = mysqli_real_escape_string($conn, $_POST['nom']);
$password = mysqli_real_escape_string($conn, $_POST['password']);


// Executer une requete SQL pour verifier les informations d'identification
$sql = "SELECT id FROM visiteurs WHERE nom = '$username' and mot_de_passe = '$password'";
$result = mysqli_query($conn, $sql);


// Verifier si la requete SQL a reussi
if ($result === false) {
    // Si la requete a echoue, afficher un message d'erreur avec les informations de
l'erreur
    echo 'Erreur SQL : ' . mysqli_error($conn);


} else {


    // Sinon, verifier si les informations d'identification sont correctes
    if (mysqli_num_rows($result) > 0) {
        // Si les informations sont correctes, rediriger vers la page protegee
        header('Location: http://localhost/Parc%20Aquatique/Interface.php');


    } else {
        // Sinon, afficher un message d'erreur
        echo 'Nom d\'utilisateur ou mot de passe incorrect';
    }
}


// Fermer la connexion a la base de donnees
mysqli_close($conn);


?>
```

Interface.php

```
<!DOCTYPE html>
<html>
  <head>
    <link rel="stylesheet" type="text/css" href="Style.css">
    <meta charset="UTF-8">
    <title>O Gliss park</title>
  </head>
  <body>
     <div class="image" style="background-image: url('FondV2.jpg');">
        <header>
          <div class="titre">

          <?php
session_start();
if (isset($_SESSION['nom'])) {
    $nom = $_SESSION['nom'];
    echo "<h1>Bienvenue Mr $nom!</h1>";
} else {
    echo "<h1>Veuillez saisir votre nom dans le formulaire précédent</h1>";
}
  ?>

          </div>
          <img src="OGliss.png" alt="Logo" style="float: left; margin-right: 1em; width:
227px; height: 90px;">
        </header>
        <nav>
          <!-- Mettre les liens de navigation ici -->
        </nav>
        <main>


        <form method="post" action="get_info.php">

          <ul>
            <li class="image1">
              <a href="get_info.php?nom=1">
                <img src="Casier.png" alt="Casier" height="100px" width="100px">
              </a>
            </li>
            <li class="image1">
              <a href="get_info.php?id=2">
                <img src="Photo.png" alt="Photos" height="100px" width="100px">
              </a>
            </li>
            <li class="image1">
              <a href="get_info.php?id=3">
                <img src="Activité.png" alt="Activites refusees" height="100px"
width="100px">
              </a>
            </li>
          </ul>
        </main>
        <footer>
          <p>Respect des normes RGPD : Ce site web respecte les normes de protection de la
vie privee et de la protection des donnees personnelles conformement au RGPD.</p>
        </footer>
      </div>
    </form>
```

SQL :

Image

```sql
-- phpMyAdmin SQL Dump
-- version 4.5.4.1
-- http://www.phpmyadmin.net
--
-- Client :  localhost
-- Généré le :  Lun 22 Mai 2023 à 11:28
-- Version du serveur :  5.7.11
-- Version de PHP :  5.6.18


SET SQL_MODE = "NO_AUTO_VALUE_ON_ZERO";
SET time_zone = "+00:00";


--
-- Base de données :  `parc_aquatique`
--


-- --------------------------------------------------------


--
-- Structure de la table `image`
--


CREATE TABLE `image` (
  `Photo` longblob NOT NULL
) ENGINE=InnoDB DEFAULT CHARSET=latin1;
```

Refus

```sql
- phpMyAdmin SQL Dump
-- version 4.5.4.1
-- http://www.phpmyadmin.net
--
-- Client :  localhost
-- Généré le :  Lun 22 Mai 2023 à 11:24
-- Version du serveur :  5.7.11
-- Version de PHP :  5.6.18


SET SQL_MODE = "NO_AUTO_VALUE_ON_ZERO";
SET time_zone = "+00:00";




--
-- Base de données :  `parc_aquatique`
--


-- --------------------------------------------------------


--
-- Structure de la table `refus`
--


CREATE TABLE `refus` (
  `NomAttraction` varchar(255) DEFAULT NULL,
  `HeureRefus` datetime NOT NULL
) ENGINE=InnoDB DEFAULT CHARSET=latin1;
```

Visiteurs

```sql
-- phpMyAdmin SQL Dump
-- version 4.5.4.1
-- http://www.phpmyadmin.net
--
-- Client :  localhost
-- Généré le :  Lun 22 Mai 2023 à 11:29
-- Version du serveur :  5.7.11
-- Version de PHP :  5.6.18


SET SQL_MODE = "NO_AUTO_VALUE_ON_ZERO";
SET time_zone = "+00:00";




-- Base de données :  `parc_aquatique`
--


-- --------------------------------------------------------


--
-- Structure de la table `visiteurs`
--


CREATE TABLE `visiteurs` (
  `IdVisiteur` int(11) NOT NULL,
  `nom` varchar(255) DEFAULT NULL,
  `prenoms` varchar(255) DEFAULT NULL,
  `mot_de_passe` varchar(255) DEFAULT NULL,
  `taille` int(11) DEFAULT NULL,
  `casier` tinyint(1) DEFAULT NULL,
  `fichier_photo` varchar(255) DEFAULT NULL,
  `numero_du_casier` int(255) DEFAULT NULL,
  `ID` int(11) DEFAULT NULL,
  `NombreRefus` int(255) DEFAULT NULL
) ENGINE=InnoDB DEFAULT CHARSET=latin1;


--
-- Contenu de la table `visiteurs`
--


INSERT INTO `visiteurs` (`IdVisiteur`, `nom`, `prenoms`, `mot_de_passe`, `taille`,
`casier`, `fichier_photo`, `numero_du_casier`, `ID`, `NombreRefus`) VALUES
(1, 'Laforge', 'Yann', 'root', 180, 0, NULL, NULL, 132456, 0),
(2, 'Poiret', 'Cedric', '123456789', 130, 1, NULL, NULL, 987654, 0),
(3, 'czqcdzf', 'sqcfq', '1234', 180, NULL, NULL, NULL, 0, NULL),
(4, 'vdese', 'sqcfqgez', 'gesrg', 180, NULL, NULL, NULL, 1768642906, NULL);
```

```sql
--
-- Index pour les tables exportées
--


--
-- Index pour la table `visiteurs`
-
ALTER TABLE `visiteurs`
  ADD PRIMARY KEY (`IdVisiteur`);


--
-- AUTO_INCREMENT pour les tables exportées
--


--
-- AUTO_INCREMENT pour la table `visiteurs`
--
ALTER TABLE `visiteurs`
  MODIFY `IdVisiteur` int(11) NOT NULL AUTO_INCREMENT, AUTO_INCREMENT=5;
```

Code Accès Casier

CLecteur.cpp

```
#include "ClecteurRFID.h" // Inclusion du fichier d'en-tête "ClecteurRFID.h"
#include <iostream> // Inclusion de la bibliothèque iostream
#include <sstream> // Inclusion de la bibliothèque sstream
#include <math.h> // Inclusion de la bibliothèque math.h
#include "pigpio.h" // Inclusion du fichier d'en-tête "pigpio.h"
#include <unistd.h> // Inclusion de la bibliothèque unistd.h
#include <wiringPi.h> // Inclusion de la bibliothèque wiringPi
ClecteurRFID::ClecteurRFID()
{
    // Constructeur
}
ClecteurRFID::~ClecteurRFID()
{
    // Destructeur
}
void ClecteurRFID::fermerportspi()
{
    gpioTerminate(); // Terminaison de la bibliothèque GPIO
}
long long int ClecteurRFID::lirebadge()
{
    long long int decimalValue = 0; // Variable pour stocker la valeur décimale du badge
    int rep = 1; // Variable pour contrôler la boucle
    mfrc522.PCD_Init(); // Initialisation du lecteur RFID
    while (rep == 1) {
        if (mfrc522.isNewCardPresent() == 0) { // Détection d'un badge RFID au niveau du
lecteur
            // std::cout << "Nouveau badge detecté..." << std::endl;
            delay(1000); // Attente de 1 seconde
            if (mfrc522.PICC_ReadCardSerial()); // Lecture de la carte RFID
            uint8_t nuidPICC[5]; // Tableau pour stocker l'identifiant du badge
            for (int i = 11; i < 16; ++i) {
                nuidPICC[i - 11] = mfrc522.uid.uidByte[i]; // Copie de l'identifiant du
badge dans le tableau
            }

            printf("\n");
            for (int i = 0; i < 4; ++i) {
                decimalValue += nuidPICC[i] << (8 * i); // Conversion de l'identifiant du
badge en valeur décimale
            }
            return decimalValue; // Retour de la valeur décimale du badge
        }
    }
}
```

CLecteurRFID.h

```
#ifndef CLECTEURRFID_H
#define CLECTEURRFID_H

#include "CMFRC522.h" // Inclusion du fichier d'en-tête "CMFRC522.h"
#include <pigpio.h> // Inclusion du fichier d'en-tête "pigpio.h"

class ClecteurRFID
{
public:
    ClecteurRFID(); // Constructeur
    ~ClecteurRFID(); // Destructeur
    void fermerportspi(); // Méthode pour fermer le port SPI
    long long int lirebadge(); // Méthode pour lire le badge RFID

private:
    CMFRC522 mfrc522; // Instance de la classe CMFRC522 pour interagir avec le lecteur RFID

};

#endif // CLECTEURRFID_H
```

BaseDeDonnees.cpp

```cpp
#include "BaseDeDonnees.h" // Inclusion du fichier d'en-tête "BaseDeDonnees.h"
#include <iostream> // Inclusion de la bibliothèque iostream
#include "requete.h" // Inclusion du fichier d'en-tête "requete.h"

BaseDeDonnees::BaseDeDonnees() {}

BaseDeDonnees::~BaseDeDonnees() {}

bool BaseDeDonnees::connexion(char *server, char *user, char *password, char *database) {
    //printf(Base;;connexion())
    mysql_init(&mysql); // Initialisation de la structure MYSQL
    maBase = mysql_real_connect(&mysql, server, user, password, database, 0, 0, 0); //
Connexion à la base de données

    if (maBase == NULL) {
        return false;
    }
    return true;
}
bool BaseDeDonnees::deconnexion() {
    mysql_close(maBase); // Fermeture de la connexion à la base de données
    return true;
}
bool BaseDeDonnees::envoyerRequete(char *requete) {
    int query_state;
    query_state = mysql_query(&mysql, requete); // Envoi de la requête SQL à la base de
données
    if (query_state != 0) {
        return false;
    }
    res = mysql_store_result(maBase); // Stockage du résultat de la requête
    return true;
}
MYSQL_RES *BaseDeDonnees::GetResponse() {
    return res; // Renvoie le résultat de la requête
}
```

CBaseDeDonnees.h

```
#ifndef BASE_DE_DONNEES_H
#define BASE_DE_DONNEES_H

#include <mysql/mysql.h> // Inclusion du fichier d'en-tête "mysql.h"
#include <string> // Inclusion de la bibliothèque string

class BaseDeDonnees {
private:
    MYSQL mysql; // Structure MYSQL pour la gestion de la base de données
    MYSQL *maBase; // Pointeur vers la connexion à la base de données
    MYSQL_RES *res; // Pointeur vers le résultat d'une requête

public:
    BaseDeDonnees(); // Constructeur
    ~BaseDeDonnees(); // Destructeur
    bool connexion(char *server, char *user, char *password, char *database); // Méthode
pour établir une connexion à la base de données
    bool deconnexion(); // Méthode pour se déconnecter de la base de données
    bool envoyerRequete(char *requete); // Méthode pour envoyer une requête SQL à la base
de données
    MYSQL_RES *GetResponse(); // Méthode pour obtenir le résultat d'une requête

};

#endif // BASE_DE_DONNEES_H
```

CRequete.cpp

```cpp
#include "requete.h" // Inclusion du fichier d'en-tête "requete.h"
#include "BaseDeDonnees.h" // Inclusion du fichier d'en-tête "BaseDeDonnees.h"
#include <iostream> // Inclusion de la bibliothèque iostream
#include "CMFRC522.h" // Inclusion du fichier d'en-tête "CMFRC522.h"
#include <chrono> // Inclusion de la bibliothèque chrono

Requete::Requete(BaseDeDonnees* bdd) {
    m_bdd = bdd; // Assignation de l'objet BaseDeDonnees passé en paramètre
}

Requete::~Requete() {}

bool Requete::verifierExistenceId(long long int numerobadge) {
    char req[500];
    sprintf(req, "SELECT ID FROM visiteur WHERE ID='%lld'", numerobadge); // Création de la
requête SQL pour vérifier l'existence de l'ID dans la table "visiteur"
    printf("La requete est %s\n", req);

    if (m_bdd->envoyerRequete(req)) { // Envoi de la requête à la base de données
        MYSQL_RES* resultat = REPONSE(); // Récupération du résultat de la requête
        if (resultat != NULL) {
            MYSQL_ROW row = mysql_fetch_row(resultat); // Récupération de la première ligne
du résultat
            mysql_free_result(resultat); // Libération de la mémoire occupée par le
résultat
            if (row != NULL) {
                return true; // L'ID existe dans la base de données
            }
        }
    }
    return false; // L'ID n'existe pas dans la base de données
}
```

CRequete.h

```
#ifndef REQUETE_H
#define REQUETE_H

#include <string> // Inclusion de la bibliothèque string
#include <vector> // Inclusion de la bibliothèque vector
#include "BaseDeDonnees.h" // Inclusion du fichier d'en-tête "BaseDeDonnees.h"

class Requete {
private:
    BaseDeDonnees* m_bdd; // Pointeur vers l'objet BaseDeDonnees

public:
    Requete(BaseDeDonnees* bdd); // Constructeur prenant en paramètre un objet
BaseDeDonnees
    ~Requete(); // Destructeur
    bool verifierExistenceId(long long int numerobadge); // Méthode pour vérifier
l'existence d'un ID dans la base de données

    MYSQL_RES *REPONSE(); // Méthode pour obtenir le résultat d'une requête

};

#endif // REQUETE_H
```

main.cpp

```
#include <iostream> // Inclusion de la bibliothèque iostream
#include <unistd.h> // Inclusion de la bibliothèque unistd.h
#include "ClecteurRFID.h" // Inclusion du fichier d'en-tête "ClecteurRFID.h"
#include <wiringPi.h> // Inclusion de la bibliothèque wiringPi
#include "BaseDeDonnees.h" // Inclusion du fichier d'en-tête "BaseDeDonnees.h"
#include "requete.h" // Inclusion du fichier d'en-tête "requete.h"
#include "CMFRC522.h" // Inclusion du fichier d'en-tête "CMFRC522.h"

using namespace std;

void delay(int ms) {
#ifdef WIN32
    Sleep(ms);
#else
    usleep(ms * 1000);
#endif
}
int main(int argc, char *argv[]) {
    BaseDeDonnees bdd; // Instance de la classe BaseDeDonnees
    if (!bdd.connexion("172.29.21.52", "utilisateur", "root", "parc")) { // Connexion à la
base de données
        cerr << "Erreur de connexion à la base de données." << endl;
        return 1;
    } else {
        if (gpioInitialise() < 0) { // Initialisation du port SPI
            cerr << "Port SPI non initialisé." << endl;
        } else {
            ClecteurRFID monlecteur; // Instance de la classe ClecteurRFID
            int retourhaut = 1; // Variable pour contrôler la boucle
            while (retourhaut != 0) {
                long long int decimalValue = monlecteur.lirebadge(); // Lecture du badge
RFID
                if (decimalValue != 0) {
                    printf("Numero de badge converti en base 10: %lli\n", decimalValue);
                    Requete req(&bdd); // Instance de la classe Requete avec la base de
données en paramètre
                    bool existe = req.verifierExistenceId(decimalValue); // Vérification de
l'existence de l'ID dans la base de données

                    std::cout << "la valeur du booléen est : " << existe << std::endl;
                    if (existe == true) {
                        std::cout << "L'ID " << decimalValue << " existe dans la base de
données." << std::endl;

                    } else {
                        cerr << "L'ID " << decimalValue << " n'existe pas dans la base de
données." << endl;
                    //    break;
                    }
                }
                delay(1000); // Attente de 1 seconde
                retourhaut = 1;
            }
            monlecteur.fermerportspi(); // Fermeture du port SPI
        }
    }

    return 0;
}
```

Code Serrure

Serrure.cpp

```
#include "serrure.h"

Serrure::Serrure() {
    wiringPiSetup(); // initialiser la bibliothèque wiringPi
    pinMode(RELAY_PIN, OUTPUT); // configurer la broche GPIO comme une sortie
    relay_pin = RELAY_PIN;
}

void Serrure::unlock() {
    printf("La serrure est déverrouillée.\n");
    digitalWrite(relay_pin, HIGH); // activer le module relais (serrure déverrouillée)
    delay(LOCK_TIME); // attendre
    digitalWrite(relay_pin, LOW); // désactiver le module relais (serrure verrouillée)
    printf("La serrure est verrouillée.\n");
}
```

Serrure.h

```
ifndef SERRURE_H
#define SERRURE_H

#include <wiringPi.h>
#include <stdio.h>

#define RELAY_PIN 29 // numéro de la broche GPIO utilisée pour le module relais
#define LOCK_TIME 2000 // durée de verrouillage/déverrouillage (en millisecondes)

class Serrure {
public:
    Serrure(); // constructeur
    void unlock(); // méthode pour déverrouiller la serrure

    //void lock();

private:
    int relay_pin; // numéro de la broche GPIO utilisée pour le module relais
};

#endif
```

main.cpp

```
#include "lecteurRFID.h"
#include "requete.h"
#include <stdio.h>
#include <stdlib.h>
#include <time.h>
#include <stdbool.h>
#include <wiringPi.h>
#include "serrure.h"
#include <iostream>
using namespace std;

int main(int argc, char* argv[]) {
    CRequete mesrequetes;
    int rep = 1;
    char nomutilisateur[30];
    char prenomutilisateur[30];
    char numerodebadge[30];
    MYSQL_ROW row;

    if (gpioInitialise() < 0) {
        printf("Port SPI non initialisé");
    }
    else {
        lecteurRFID monlecteur;
        printf("Port SPI initialisé");

        srand(time(NULL)); // initialiser le générateur de nombres aléatoires
        Serrure serrure;

        while (rep == 1) {
            long long int numerobadge = monlecteur.lirebadge();
            if (numerobadge != 0) {
                printf("numero de badge convertie en base 10: %lli", numerobadge);
                printf("\n");

                cout << "Bienvenue dans la base parc pour faire du CRUD sur la table
casier" << endl;
                if (mesrequetes.connexionBdd() == true) {
                    printf("connexion ok\n");

                    printf("afficher un badge %lli \n", numerobadge);
                    if (mesrequetes.lireDonnees(numerobadge) == true) {
                        printf("requete réussie \n");

                        unsigned int num_champs = mysql_num_fields(mesrequetes.REPONSE());

                        printf("\n nombre de champs %i.\n", num_champs);

                        //Tant qu'il y a encore un résultat ...
                        while ((row = mysql_fetch_row(mesrequetes.REPONSE())))
                        {
                            //On fait une boucle pour avoir la valeur de chaque champs
                            for (int i = 0; i < num_champs; i++)
                            {
                                //On ecrit toutes les valeurs
```

```
                                printf("-%s- ", row[i]);
                            }
                            /*printf("\n");
                            strcpy(numerodebadge,row[1]);
                            strcpy(nomutilisateur,row[2]);
                            strcpy(prenomutilisateur,row[3]);
                            printf("\n Badge : %s, , %s  , %s
\n",numerodebadge,nomutilisateur,prenomutilisateur);
                            */
                        }

                        //Libération du jeu de résultat
                        mysql_free_result(mesrequetes.REPONSE());
                    }
                }
                printf("Voulez vous continuer ? oui:1 non:0\n");
                scanf("%i", &rep);
                mesrequetes.deconnexionBdd();
            }

            int random_number = rand() % 2; // générer un nombre aléatoire entre 0 et 1
            if (random_number == 0) {
                serrure.unlock(); // déverrouiller la serrure
            }
            delay(5000); // attendre 5 secondes avant de vérifier à nouveau
        }

        monlecteur.fermerportspi();
    }

    return 0;
}
```

Code Prise Photo

Main.cpp

```
#include <stdio.h>
#include <stdlib.h>
#include <wiringPi.h>

#define TRIG_PIN 15
#define ECHO_PIN 16

int main() {
    wiringPiSetup();
    pinMode(TRIG_PIN, OUTPUT);
    pinMode(ECHO_PIN, INPUT);

    while (1) {
        digitalWrite(TRIG_PIN, LOW);
        delayMicroseconds(2);
        digitalWrite(TRIG_PIN, HIGH);
        delayMicroseconds(10);
        digitalWrite(TRIG_PIN, LOW);

        while (digitalRead(ECHO_PIN) == LOW);
        long start_time = micros();

        while (digitalRead(ECHO_PIN) == HIGH);
        long end_time = micros();

        long travel_time = end_time - start_time;
        int distance = travel_time / 58;

        printf("Distance: %dcm\n", distance);

        if (distance < 30) {
            char command[] = "gphoto2 --capture-image-and-download --filename
/home/pi/Images/photo.jpg";
            system(command);
            break;
        }

        delay(1000);
    }

    return 0;
}
```