



# LA RUCHE



## Equipe :

- **BEDNARICK**  
thomas
- **LEIROS**  
Mateo
- **TAMBOISE**  
Paul

# SOMMAIRE

## Partie commune :

- Contexte
- Expression du besoin
- Impact environnemental

## Partie personnelle du projet :

- Analyses et solutions techniques retenues
- Analyses et langages retenus
- Programmes développés
  - Page Web
  - CSS
  - Arduino

## Conclusion

# CONTEXTE

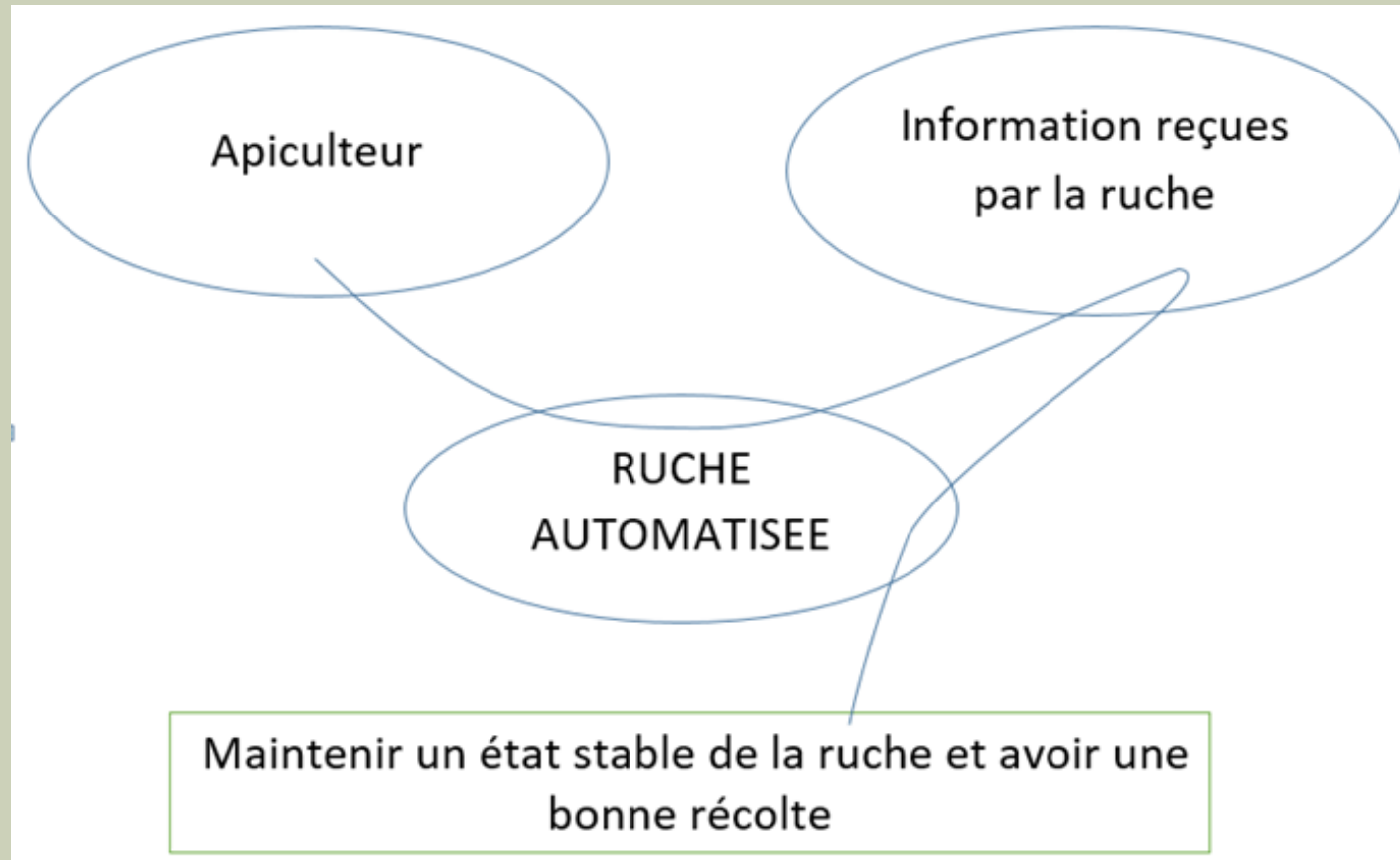
## **Problématique :**

- Comment surveiller à distance la santé des abeilles et la quantité de miel produite dans la ruche et communiquer ces informations à l'apiculteur.

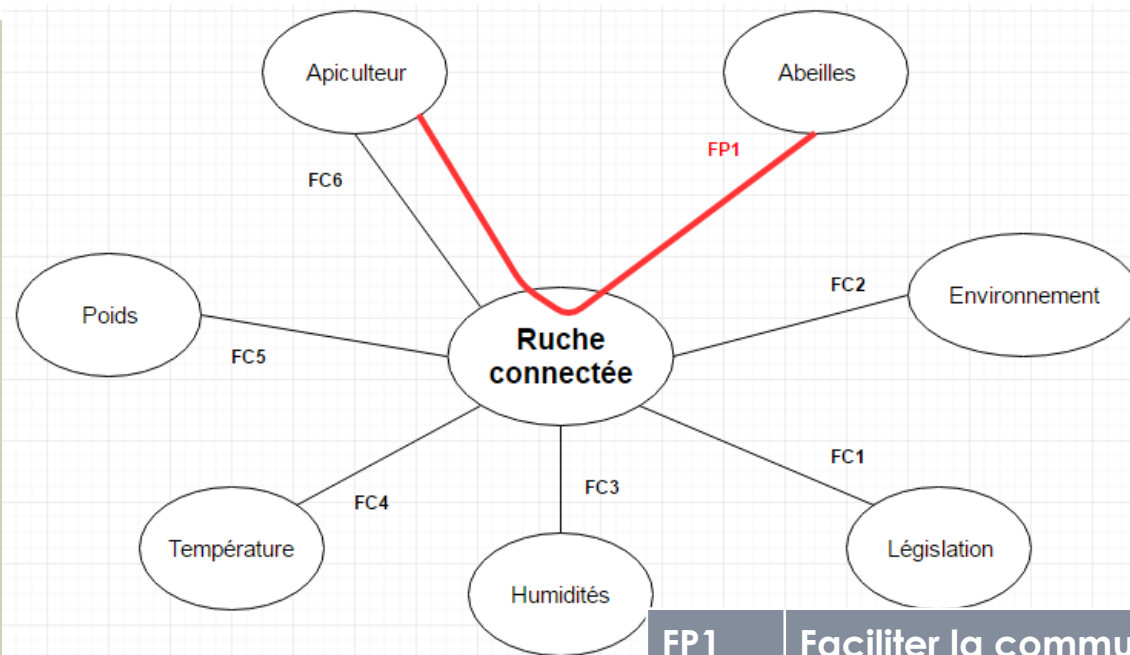
## **Fonctions principales attendues :**

- Surveiller la température
- Surveiller l'humidité
- Surveiller le poids
- Transférer et afficher les données
- Respect du développement durable

# EXPRESSION DU BESOIN



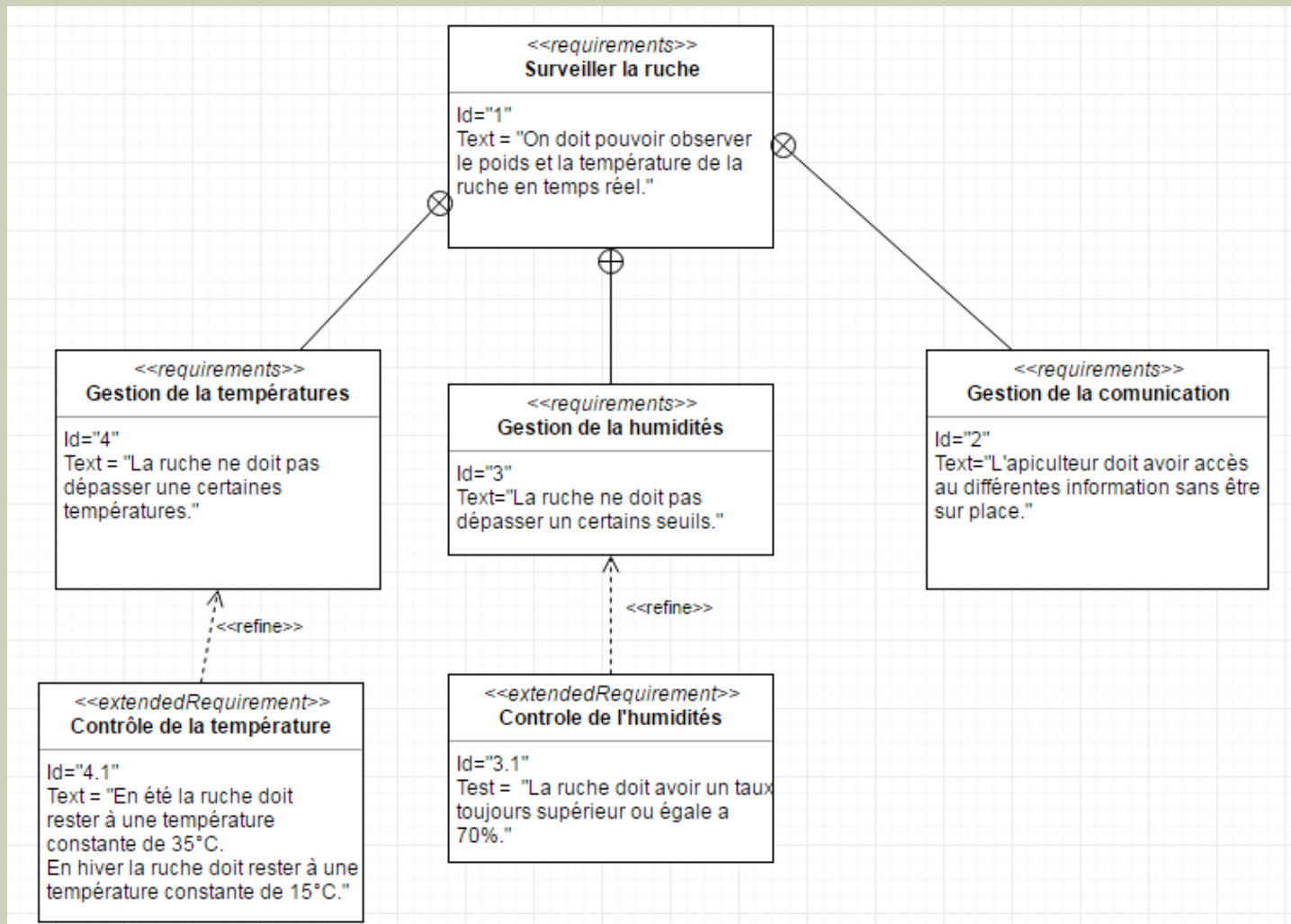
# EXPRESSION DU BESOIN



FP1	Faciliter la communication entre la ruche et l'apiculteur
FC1	Respecter les lois imposées
FC2	Respecter le coté écologique
FC3	Mesurer la température de la ruche
FC4	Mesurer la température de la ruche
FC5	Mesurer le poids de la ruche
FC6	Récolter le miel

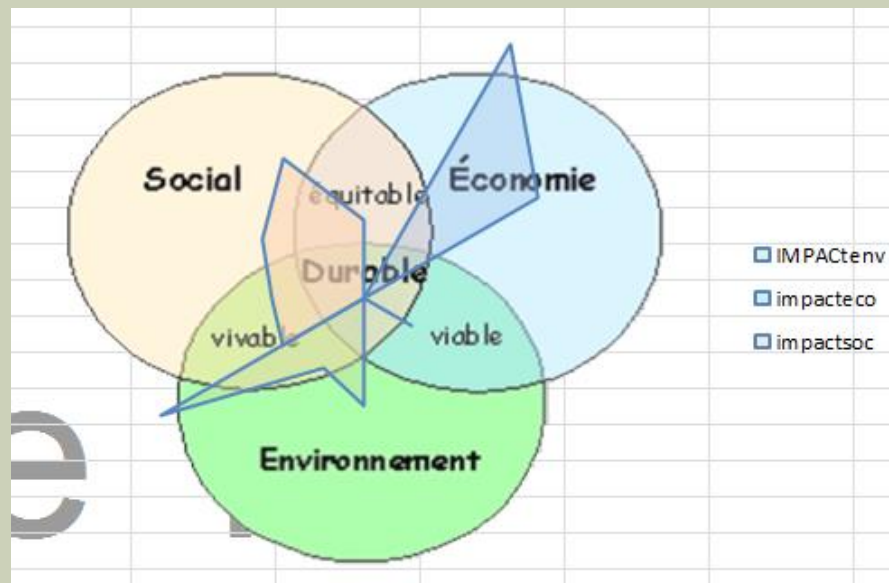
# EXPRESSION DU BESOIN

## DIAGRAMME D'EXIGENCE

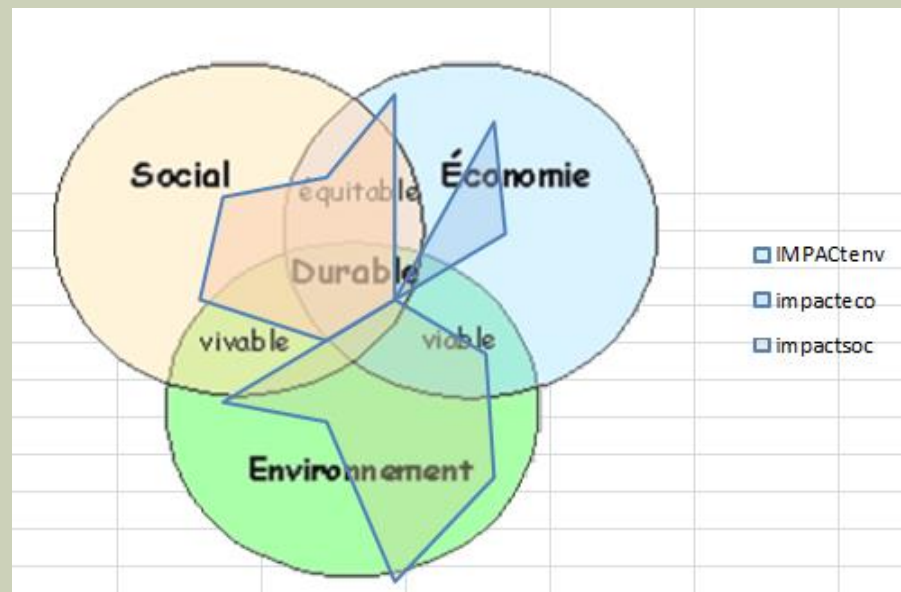


# IMPACT ENVIRONNEMENTAL

Avant



Après



# PARTIE PERSONNELLE DU PROJET

## Problématique :

Créer une interface homme machine, la stocker et afficher des informations dans un réseau local.

Microcontrôleur

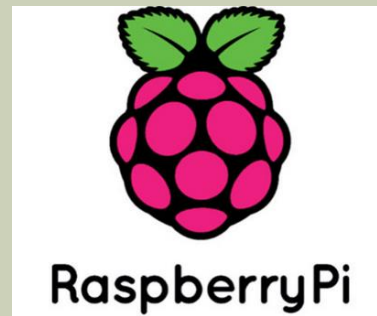
Fonctions de  
communication

Affichage



# MICROCONTRÔLEUR

## Solutions techniques étudiées :



## Solution retenue, ARDUINO :

- ✓ Langage de programmation simple
- ✓ Open – Source
- ✓ Coût financier réduit
- ✓ Communauté riche (sur internet)



# CHOIX DE LA CARTE ARDUINO

## ARDUINO UNO

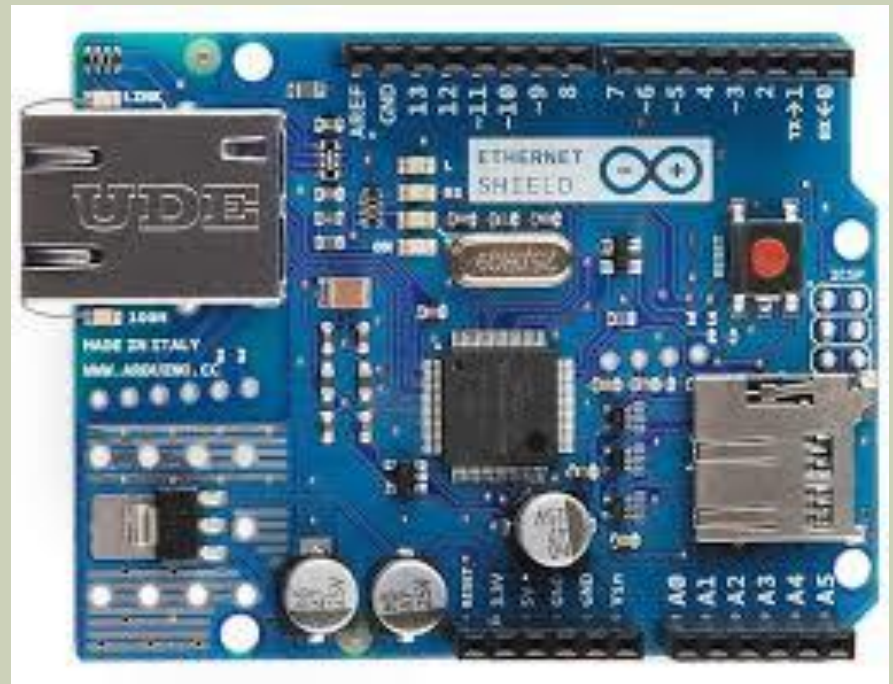


## ARDUINO MEGA



# COMPOSANTS UTILISÉS

- Lecteur de carte SD
- Puce Wiznet W5100
- Port Rj45
- Full-duplex
- Bus SPI
- Communication:
  - Pins 10, 11, 12, 13.
  - Pins 4 carte SD.



# INFORMATION : BUS SPI

- **Fonctions principales:**

- SCLK : Serial Clock, Horloge (généré par le maître)
- MOSI : Master Output, Slave Input (généré par le maître)
- MISO : Master Input, Slave Output (généré par l'esclave)
- SS : Slave Select, Actif à l'état bas (généré par le maître)

# LANGAGES ANALYSES

- HTML :

- Programmation simple, compatibilité élevée et programmation via Arduino.



- PHP:

- Complexe à mettre en œuvre, lourd, cependant offre de nombres fonctions.



- Le Python :

- Léger, programmation simple, mais options limitées.



# LANGAGES RETENUS

## HTML (Hyper Text Markup Language) :

- Créer des pages web (.html)
- Système de balises
- Éditeur de texte (Notepad ++)
- Langage simple
- Langage de programmation

## CSS (Cascading Style Sheet) :

- Le CSS est utilisé pour la présentation du site (Police, couleur, arrière-plan, ...).
- Alléger le HTML
- Style lourd et complexe



# PROGRAMMATION PAGE WEB

```
1 <!DOCTYPE html>
2 <html>
3 <head>
4     <meta charset="utf-8" />
5     <link rel="stylesheet" href="styleV1.css"/>
6     <title> Premiers tests du site </title>
7 </head>
8
9 <body>
10
11
12 <header>
13     <h1>Votre ruche</h1>
14 <nav>
15
16 <ul>
17     <li><a href="index.html">Accueil</a></li>
18     <li><a href="mailto:mateoleiros@gmail.com">Assistance</a></li>
19 </ul>
20 </nav>
21 </header>
22
23 <section>
24     
25     <h1> <strong> <em> Créateur </em> </strong></h1>
26     <p>Prénom : Mateo </br> Nom : Leiros </br> Classe : Tsin2 </br> </p>
27     <p>Bonjour, dans le cadre de mon projet de Terminale STI2D option SIN.</br>Je dois créer un site web pour répertorier les informations obtenu
28 </section>
29 <aside>
30     <h1><strong> Voici les différentes variables <strong> </h1>
31     <iframe src="po.txt" <a frameborder="0">
32         </iframe>
33 </aside>
34
```

# PROGRAMMATION CSS

```
1 header
2 {
3     text-align: center;
4     color: red;
5     font-size: x-large;
6     text-decoration: underline;
7     font-family: georgia;
8
9 }
10
11 body
12 {
13     background-image: url("clair.jpg");
14     background-repeat: no-repeat;
15     background-size: cover;
16     text-align: center;
17     font-family: arial;
18
19 }
20
21
22
23
24 aside
25 {
26     font-family: arial;
27     text-align: justify;
28     margin: 50px;
29     background-attachment: fixed;
30     font-size: x-large;
31
32 }
33
```

```
34 nav
35 {
36     font-family: arial;
37     display: inline-block;
38     width: 740px;
39     text-align: right;
40     position: absolute;
41     right: 70px;
42     top: 20px;
43     font-weight: bold;
44 }
45 nav ul
46 {
47     list-style-type: none;
48 }
49
50 nav li
51 {
52     display: inline-block;
53     margin-right: 15px;
54 }
55 strong
56 {
57     font-size: large;
58 }
59
60 section
61 {
62     text-align: justify;
63 }
64
65 p
66 {
67     text-shadow: 2px 2px yellow;
68 }
69
70
```



# PROGRAMMATION

## PREMIER PROGRAMME ARDUINO

```
#include <SPI.h>
#include <Ethernet.h>
byte mac[] = {0x90, 0xA2, 0xDA, 0x0F, 0xDF, 0xAB};
byte ip[] = {192, 168, 1, 105};
EthernetServer serveur(80);

void setup() {
  Serial.begin(9600);
  Ethernet.begin(mac, ip);
  Serial.print("\nLe serveur est sur l'adresse : ");
  Serial.println(Ethernet.localIP());
  serveur.begin();
}

void loop() {
  EthernetClient client = serveur.available();
  if (client) {
    Serial.println("Client en ligne\n");
    if (client.connected()) {
      while (client.available()) {
        char c=client.read();
        Serial.write(c);
        delay(1);
      }
    }
  }
}
```

Adresses et bibliothèques

Préparation à la connexion

Vérification présence  
utilisateur

# PROGRAMMATION

## PREMIER PROGRAMME ARDUINO

```
client.println("<!DOCTYPE HTML>");
client.println("<html>");
client.println("<head>");
client.println("<meta charset=\"utf-8\" />");
client.println("<title> Premiers tests du site </title>");
client.println("</head>");
client.println("<body>");
client.println("<header>");
client.println("<h1>La Ruche </h1>");
client.println("<hr>");
client.println("<section>");
client.println("<h1> <strong> Créateur </strong></h1>");
client.println("<p>Prénom : Matéo </br> Nom : Leiros </br> Classe");
client.println("<p>Bonjour, dans le cadre de mon projet de Termin");
client.println("<aside>");
client.println("<h1> <strong> Voici les différentes variables <st");
client.println("</aside>");
client.println("</section>");
client.println("</body>");
client.println("</header>");
client.println("</html>");
client.stop();
Serial.println("Fin de communication avec le client");
}
```

Codage site Web

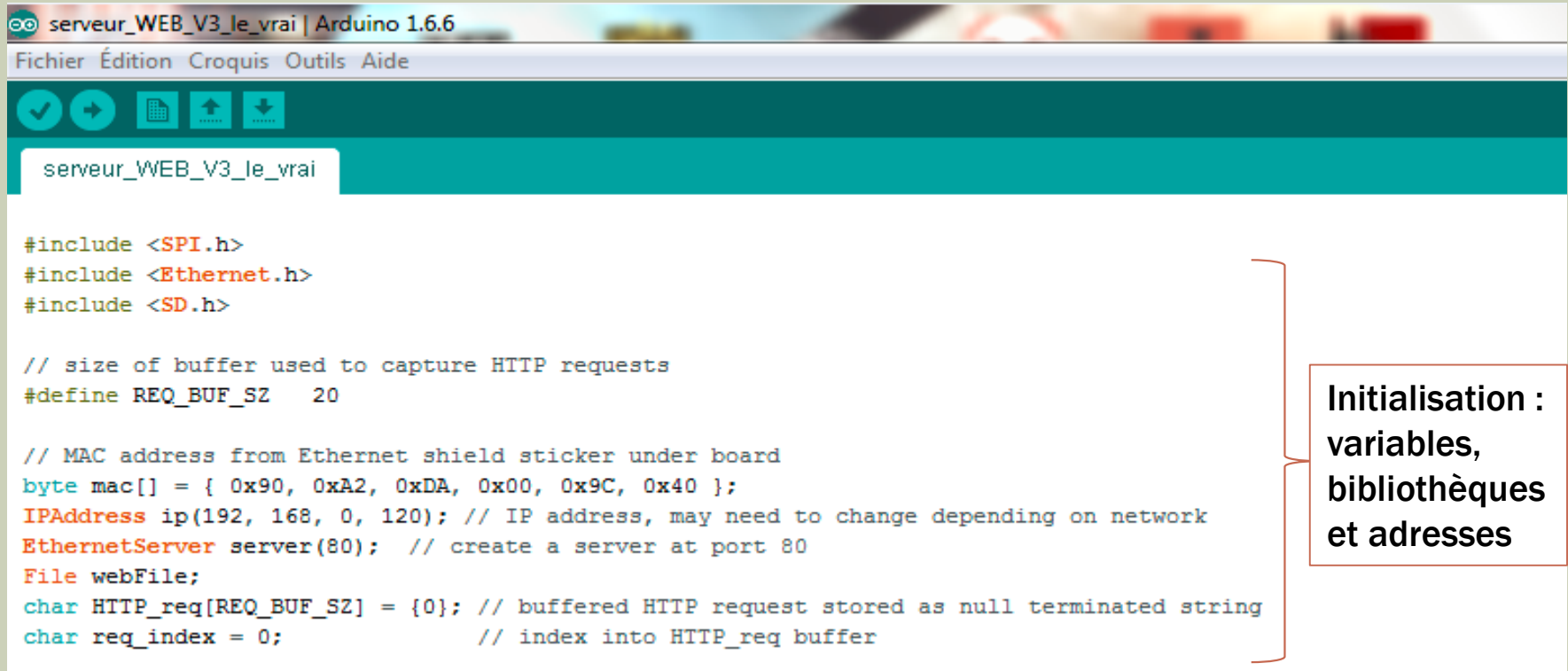
# PROGRAMMATION

# PREMIER PROGRAMME



# PROGRAMMATION

## SECOND PROGRAMME ARDUINO



```
serveur_WEB_V3_le_vrai | Arduino 1.6.6
Fichier Édition Croquis Outils Aide

serveur_WEB_V3_le_vrai

#include <SPI.h>
#include <Ethernet.h>
#include <SD.h>

// size of buffer used to capture HTTP requests
#define REQ_BUF_SZ 20

// MAC address from Ethernet shield sticker under board
byte mac[] = { 0x90, 0xA2, 0xDA, 0x00, 0x9C, 0x40 };
IPAddress ip(192, 168, 0, 120); // IP address, may need to change depending on network
EthernetServer server(80); // create a server at port 80
File webFile;
char HTTP_req[REQ_BUF_SZ] = {0}; // buffered HTTP request stored as null terminated string
char req_index = 0; // index into HTTP_req buffer
```

Initialisation :  
variables,  
bibliothèques  
et adresses

# PROGRAMMATION

## SECOND PROGRAMME ARDUINO

```
void setup()
{
    // disable Ethernet chip
    pinMode(10, OUTPUT);
    digitalWrite(10, HIGH);

    Serial.begin(9600);      // for debugging

    // initialize SD card
    Serial.println("Initializing SD card...");
    if (!SD.begin(4)) {
        Serial.println("ERROR - SD card initialization failed!");
        return;    // init failed
    }
    Serial.println("SUCCESS - SD card initialized.");
    // check for index.htm file
    if (!SD.exists("index.htm")) {
        Serial.println("ERROR - Can't find index.htm file!");
        return;    // can't find index file
    }
    Serial.println("SUCCESS - Found index.htm file.");

    Ethernet.begin(mac, ip); // initialize Ethernet device
    server.begin();          // start to listen for clients
}
```

Préparation à la connexion et  
communication avec la carte SD

# PROGRAMMATION

## SECOND PROGRAMME ARDUINO

```
void loop()
{
    EthernetClient client = server.available(); // try to get client
    if (client) { // got client?
        boolean currentLineIsBlank = true;
        while (client.connected()) {
            if (client.available()) { // client data available to read
                char c = client.read(); // read 1 byte (character) from client
                // buffer first part of HTTP request in HTTP_req array (string)
                // leave last element in array as 0 to null terminate string (REQ_BUF_SZ - 1)
                if (req_index < (REQ_BUF_SZ - 1)) {
                    HTTP_req[req_index] = c;          // save HTTP request character
                    req_index++;
                }
                // print HTTP request character to serial monitor
                Serial.print(c);
                // last line of client request is blank and ends with \n
                // respond to client only after last line received
                if (c == '\n' && currentLineIsBlank) {
                    // open requested web page file
                    if (StrContains(HTTP_req, "GET / ")
                        || StrContains(HTTP_req, "GET /index.htm")) {
                        client.println("HTTP/1.1 200 OK");
                        client.println("Content-Type: text/html");
                        client.println("Connnection: close");
                        client.println();
                        webFile = SD.open("index.htm");          // open web page file
                    }
                }
            }
        }
    }
}
```

Vérification présence  
utilisateur

Récupération,  
transmission page  
Web

# PROGRAMMATION

## SECOND PROGRAMME ARDUINO

```
    }  
else if (StrContains(HTTP_req, "GET /abeille.jpg")) {  
    webFile = SD.open("abeille.jpg");  
    if (webFile) {  
        client.println("HTTP/1.1 200 OK");  
        client.println();  
    }  
}  
  
else if (StrContains(HTTP_req, "GET /styleV1.css")) {  
    client.println("HTTP/1.1 200 OK");  
    client.println("Content-Type: text/css");  
    client.println("Connnection: close");  
    client.println();  
    webFile = SD.open("styleV1.css");           // open web page file  
}  
else if (StrContains(HTTP_req, "GET /clair.jpg")) {  
    webFile = SD.open("clair.jpg");  
    if (webFile) {  
        client.println("HTTP/1.1 200 OK");  
        client.println();  
    }  
}  
else if (StrContains(HTTP_req, "GET /po.txt")) {  
    webFile = SD.open("po.txt");  
    if (webFile) {  
        client.println("HTTP/1.1 200 OK");  
        client.println();  
    }  
}
```

Récupération,  
transmission image

Récupération,  
transmission CSS

Récupération,  
transmission image

Récupération,  
transmission fichier  
texte

# PROGRAMMATION

## SECOND PROGRAMME ARDUINO

```
    }
    if (webFile) {
        while(webFile.available()) {
            client.write(webFile.read()); // send web page to client
        }
        webFile.close();
    }
    // reset buffer index and all buffer elements to 0
    req_index = 0;
    StrClear(HTTP_req, REQ_BUF_SZ);
    break;
}
// every line of text received from the client ends with \r\n
if (c == '\n') {
    // last character on line of received text
    // starting new line with next character read
    currentLineIsBlank = true;
}
else if (c != '\r') {
    // a text character was received from client
    currentLineIsBlank = false;
}
} // end if (client.available())
} // end while (client.connected())
delay(1); // give the web browser time to receive the data
client.stop(); // close the connection
} // end if (client)
```

Clôture de la  
communication



# PROGRAMMATION

## SECOND PROGRAMME ARDUINO

```
void StrClear(char *str, char length)
{
    for (int i = 0; i < length; i++) {
        str[i] = 0;
    }
}

char StrContains(char *str, char *sfind)
{
    char found = 0;
    char index = 0;
    char len;
    len = strlen(str);
    if (strlen(sfind) > len) {
        return 0;
    }
    while (index < len) {
        if (str[index] == sfind[found]) {
            found++;
            if (strlen(sfind) == found) {
                return 1; }
        }
        else {
            found = 0;
        }
        index++;
    } return 0;
}
```

# PROGRAMMATION


## DEUXIEME PROGRAMME

Premiers tests du site x

192.168.0.120

# Votre ruche

[Accueil](#) [Assistance](#)



**Créateur**

Prénom : Mateo  
Nom : Leiros  
Classe : Tsin2

Bonjour, dans le cadre de mon projet de Terminale STI2D option SIN.  
Je dois créer un site web pour répertorier les informations obtenues depuis mon arduino

**Voici les différentes variables**

Temperature : 35  
Humidites : 70

# CONCLUSION

## MERCI DE VOTRE ATTENTION

### Equipe :

- **BEDNARICK**  
thomas
- **LEIROS**  
Mateo
- **TAMBOISE**  
Paul