**Une image contenant flèche

Description générée automatiquementUne image contenant texte, clipart

Description générée automatiquementUne image contenant texte, clipart

Description générée automatiquement**

Documentation Technique

Version 1.0

**Auteur :** Jérémie Cyr

**Professeur :** Alain Dubé

**Date :** 14 septembre 2021

**Table des matières**

[**Demande** 3](#_Toc89174373)

[**Client et contacts** 3](#_Toc89174374)

[**Besoins** 3](#_Toc89174375)

[**Proposition** 4](#_Toc89174376)

[**Conception** 5](#_Toc89174377)

[**Contexte** 5](#_Toc89174378)

[**Technologies** 5](#_Toc89174379)

[**Le microcontrôleur (ESP32) Crédits : Alain Dubé** 5](#_Toc89174380)

[**Écran OLED I2C 0.96 pouce (128x64) Crédits : Alain Dubé** 5](#_Toc89174381)

[**Senseur de température et d’humidité (DHT22)** 6](#_Toc89174382)

[**Librairies à utiliser** 6](#_Toc89174383)

[**Environnement de simulation** 6](#_Toc89174384)

[**Preuves de concept à planifier** 6](#_Toc89174385)

[**Liste des preuves de concepts prévues** 7](#_Toc89174386)

[**Planification** 7](#_Toc89174387)

[**Mise en œuvre** 9](#_Toc89174388)

[**Preuves de concept** 9](#_Toc89174389)

[**DHT22** 9](#_Toc89174390)

[**Branchement sur le ESP** 9](#_Toc89174391)

[**Librairies utilisées** 9](#_Toc89174392)

[**Particularités rencontrées lors de la programmation** 10](#_Toc89174393)

[**Code** 10](#_Toc89174394)

[**Analyse et conclusion** 10](#_Toc89174395)

[**Annexes** 11](#_Toc89174396)

# **Demande**

## **Client et contacts**

**Les Meubles Dubé inc**

**Travail de session A2021**

**Client : M. Alain Dubé**

## **Besoins**

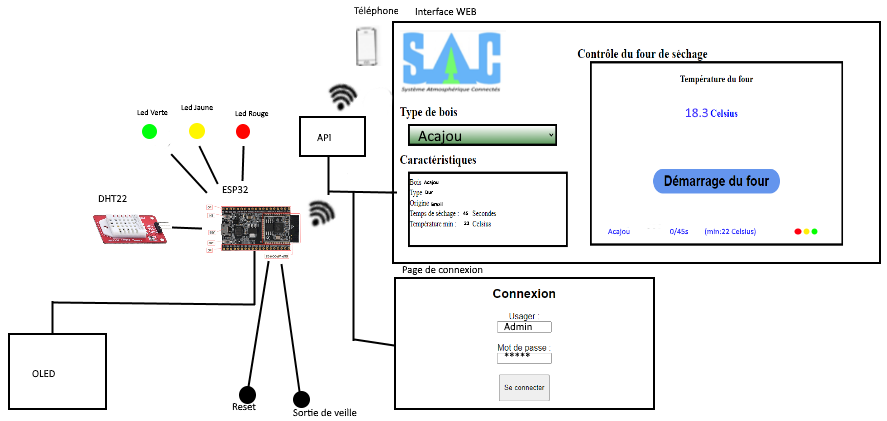
**Dans ce projet, nous voulons répondre à la demande d’un client qui souhaite avoir un Système Atmosphérique Connecté (SAC) pour son four de séchage dans son usine. Le système doit contrôler la température et le temps de séchage du bois selon le type du bois. Chacun des fours sont connectés à un service WEB pour choisir le bois à traiter et afficher le temps de séchage et l’état du séchage.**

**Voici les fonctionnalités désirées par le client :**

* **Pouvoir sélectionner un type de bois en récupérant les informations par le biais d’un API connecté à une base de données. Tout ça dans une interface WEB identique à celle fournie.**
* **Pouvoir appuyer sur un bouton pour démarrer la séquence de séchage une fois le type de bois choisis et le bois inséré dans le four.**
* **Un compte à rebours qui démarre une fois le bouton pressé si la température est la bonne, sinon le système se met en mode attente et attend d’être à la bonne température pour démarrer le compte à rebours.**
* **Le système doit configurer la sensibilité des boutons tactiles au démarrage.**
* **Le système doit se connecter au Wifi, s’il ne connait pas le réseau, il demande l’accès en utilisant un Access Point.**
* **Quand le système est connecté, les leds Rouge / Vert / Jaune s’allument et s’éteignes deux fois.**
* **Quand le système ne fait rien, il se met en mode attente.**
* **Le système indique heating pendant que le four est à la bonne température et que le compte à rebours compte.**

## **Proposition**

**Une solution pas chère mais efficace de créer un système comme ça, c’est d’utiliser un module ESP32. Il a l’avantage de consommer peu d’énergie, est très petit, ne coûte pas cher et possède son propre point d’accès sans fil. Voici un schéma qui démontre le système que nous proposons.**

****

**Comme nous pouvons le constater dans le schéma, le ESP peut être configuré en point d’accès sans fil permettant ainsi de pouvoir configurer les paramètres nécessaires au système. Lorsque la page HTML de connexion sera affichée, l’utilisateur pourra entrer l’identifiant et le mot de passe pour accéder à la page de gestion du four. Ces pages sont inscrites directement dans le microcontrôleur.**

**Pour connaître la température du four, nous utiliserons un senseur de température DHT22. Il fonctionne à 3.3v et a besoin de peu d’ampérage. L’ESP pourra facilement l’utiliser directement.**

**Le système pourra afficher des informations en temps réels sur un écran OLED. Il y a plusieurs interfaces différentes qui comportes les informations sur le système, la température actuelle, etc. L’interface changera selon le mode dans lequel le four se trouve.**

**Finalement, nous allons créer une API et un serveur Web embarquée qui pourra être utilisée par le système connecté. Le client pourra s’il le désire utiliser son téléphone intelligent pour accéder à l’interface.**

# **Conception**

## **Contexte**

**Travail de session**

## **Technologies**

**Voici une liste des technologies utilisées dans le fonctionnement du système.**

### **Le microcontrôleur (ESP32) Crédits : Alain Dubé**

**Une image contenant texte, équipement électronique

Description générée automatiquementLe microcontrôleur utilisé ici est le ESP32 construit par la compagnie Espressif. Voici les spécifications importantes pour nous :**

* + - * **Processeur: microprocesseur Xtensa double cœur 32 bits**
      * **Voltage d’opération : 3.3V (contrairement à l’Arduino qui est de 5V)**
      * **Plus de 16 I/O broches**
      * **Entrée analogue : 1**
      * **Mémoire flash : 4Mb**
      * **SRAM : 520 Kb**
      * **Vitesse : 80 ou 240 Mhz**
      * **Wifi, Bluetooth, « Touch sensor »(Hall)**
      * **SPI (4x), I2C (2x), UART (3x)**
      * **Prix : environ 10$**

**Référence :**

* [**Premiers tests pour bien débuter avec l'ESP32 [ESP32 #1].**](https://www.youtube.com/watch?v=zqwnYuOLvsE)

### **Écran OLED I2C 0.96 pouce (128x64) Crédits : Alain Dubé**

**Une image contenant texte, équipement électronique

Description générée automatiquementCet écran OLED permet de facilement afficher des informations.**

* + - * **I2C**
      * **3.3v / 5v**

**Référence :**

* [**Playing Videos on 128X64 OLED Display with ESP32 | Bahubali Movie on OLED Display**](https://www.youtube.com/watch?v=9duliwzgnV4)
* [**Lien moodle**](https://rdl.moodle.decclic.qc.ca/mod/page/view.php?id=35280)

### **Senseur de température et d’humidité (DHT22)**

** Ce senseur de température et d’humidité permet d’obtenir facilement la température ou l’humidité de l’air ambiant.**

* + **Pin 1 : VDD : 3.3v**
  + **Pin 2 : DATA : signal**
  + **Pin 3 : NULL**
  + **Pin 4 : GND**

**Référence :**

* [**DHT22.pdf (stackpathcdn.com)**](http://e6n3v2t9.stackpathcdn.com/wp-content/uploads/2021/01/DHT22.pdf)

## **Librairies à utiliser**

**lib\_deps =**

* **DHT**

## **Environnement de simulation**

**Il n’y a pas de four de séchage ni de bois à disposition, mais l’environnement est suffisant pour faire les preuves de concepts et le prototype.**

## **Preuves de concept à planifier**

**Pour prouver notre solution, on se doit de vérifier si nos technologies choisies sont les bonnes et si nous réussirons à les implémenter. Pour en être certain, nous allons tester chacune d’entre elles.**

## **Liste des preuves de concepts prévues**

|  |  |  |
| --- | --- | --- |
| **Technologie** | **Temps nécessaire** | **Commentaires** |
| **DHT22** | **160 minutes** | **Prise en main et test pour vérifier si le senseur capte bien la température**  **Liens :** [**lien1**](https://rdl.moodle.decclic.qc.ca/course/view.php?id=1174)**,** [**lien2**](http://e6n3v2t9.stackpathcdn.com/wp-content/uploads/2021/01/DHT22.pdf) |
| **Total** | **160 minutes** |  |

# **Planification**

**Il est important de bien planifier le travail à faire pour rentrer dans notre temps et notre budget. Voici ce que nous prévoyons faire pour ce projet.**

|  |  |  |
| --- | --- | --- |
| ÉLÉMENT | Temps  (minutes) | Date |
| **Préparer un environnement de programmation fonctionnel**  **Ce n’est pas la première fois que nous programmons sur un ESP32. Nous allons installer Platform IO et allons-nous assurer que nous sommes capables de transférer du code à l’intérieur.** | **60** | **1 novembre** |
| **Préparer l’environnement de simulation** | **30** | **1 novembre** |
| **Nous allons trouver une façon de faire varier la température pour pouvoir simuler la température à l’intérieur d’un four de séchage.** |  |  |
| **Installer un système de versions** | **15** | **1 novembre** |
| **Nous allons créer un répertoire pour notre projet et allons installer Git avec un « repository » externe GitHub.** |  |  |
| **Programmer l’interface graphique Web pour le four** | **75** | **1-2 novembre** |
| **Nous allons programmer une interface Web qui n’est pour l’instant pas connectée à une api mais qui passe par l’ESP32 pour le routage.** |  |  |
| **Programmer l’API pour obtenir les informations de la BD** | **90** | **2 novembre** |
| **Il sera possible de configurer le système à partir d’une page WEB. Cette page est directement accessible par le ESP32 qui devient un point d’accès.** |  |  |
| **Programmer la preuve de concept : DHT22** | **30** | **2 novembre** |
| **Voir la liste des preuves de concepts dans la section précédente.** |  |  |
| **Programmer les interactions avec les composantes** | **120** | **2-8 novembre** |
| **Senseur de température/écran OLED** |  |  |
| **Programmer l’interface utilisateur** | **90** | **8 novembre** |
| **Finaliser les interfaces WEB et Oled pour l’interraction avec l’ESP.** |  |  |
| **Sécuriser l’API** | **180** | **9 novembre** |
| **Sécuriser l’API avec un système de token.** |  |  |
| **Modifier l’interface Web pour ajouter la connexion sécurisée en utilisant les tokens** | **180** | **15 novembre** |
| **Ajouter une page de connexion qui valide avec l’API en passant par l’ESP que les identifiants sont valides et récupère le token envoyé par l’API pour avoir l’autorisation d’effectuer des requêtes à l’API** |  |  |
| **Contrôler la qualité de l’application** | **180** | **16 novembre** |
| **Ce qui veut dire créer des plans de tests fonctionnels, faire la revue de code, s’assurer que le produit est conforme aux besoins du client et que la gestion des versions est optimale.** |  |  |
| **Préparer la documentation pour le client** | **60** | **22 novembre** |
| **Manuel de l’utilisateur** |  |  |
| **Livrer le produit au client** | **60** | **22 novembre** |
| **Ce qui veut dire de déployer et installer le système pour le client.** |  |  |
| **Rédiger la documentation technique** | **180** | **22-23 novembre** |
| **Finaliser le manuel technique (informations pertinentes et claires).** |  |  |
| **Temps réservé si besoin de temps supplémentaire** | **420** | **23-29-30 novembre** |
| **Remise Finale** |  | **1 décembre** |

# **Mise en œuvre**

## **Preuves de concept**

**Pour réaliser le projet et faire les preuves de concept, il faut nous procurer certains éléments dont nous ne disposons pas.**

[**DHT22**](https://www.amazon.ca/-/fr/capteur-temp%C3%A9rature-dhumidit%C3%A9-Arduino-Raspberry/dp/B07CM2VLBK/ref=sr_1_5?__mk_fr_CA=%C3%85M%C3%85%C5%BD%C3%95%C3%91&dchild=1&keywords=dht22&qid=1634675143&sr=8-5)** (16.49$)**

## **DHT22**

### **Branchement sur le ESP**

**Une image contenant texte, équipement électronique, circuit

Description générée automatiquement**

### **Librairies utilisées**

**DHT**

### 

### **Particularités rencontrées lors de la programmation**

* **Utiliser TemperatureStub fourni par le professeur**
* **Inclure le .h suivant :**

**#include "temperature/TemperatureStub.h"**

* **Dans notre cas il faut définir :**

**#define DHTPIN 15 // Pin utilisée par le senseur DHT22**

**#define DHTTYPE DHT22 //Le type de senseur utilisé**

* **Pour initialiser :**

**TemperatureStub \*temperatureStub = NULL;**

### **Code**

**Voir le projet : senseurTemp**

### **Analyse et conclusion**

**Le senseur de température fonctionne bien. Il répond aux besoins de notre système. Il est compact, facile d’utilisation et assez précis.**

# **Annexes**