06/10/2025

Veronique Cloutier & Félix-Alexandre Morneau-Carrier

Cégep Limoilou

Projet – partie 3

Distributeur automatique de nourriture avec reconnaissance

Table des matières

[Introduction 1](#_Toc199173020)

[Schéma 2](#_Toc199173021)

[Interface et base de données 3](#_Toc199173022)

[Problèmes rencontrés 4](#_Toc199173023)

[Annexe 1 : Queries SQL pour la création des tables et l’insertion de données test 5](#_Toc199173024)

# Introduction

Ce projet vise à concevoir un distributeur de nourriture automatique et intelligent pour chats, basé sur les technologies IoT. L’objectif est de contrôler l’accès à la nourriture à l’aide de la reconnaissance RFID, afin que seuls les chats autorisés puissent manger. Le système détecte la présence de l’animal, vérifie son identité via une puce RFID, et ouvre automatiquement le compartiment à l’aide d’un servomoteur.

Le présent document est le troisième d’une série de quatre, et il concerne particulièrement la partie interface du projet. Seront donc présentés un schéma des connexions entre les différents éléments physiques, puis on présentera notre interface en la documentant, suite à quoi nous présenterons notre base de données. Finalement, les problèmes rencontrés seront discutés.

# Schéma

Une image contenant texte, capture d’écran, Police, diagramme

Le contenu généré par l’IA peut être incorrect.

**Légende :**

Flèches bleues = requêtes

Flèches oranges = réponses

# Interface et base de données

Nous voulions que l’utilisateur puisse utiliser une interface facilement accessible en tout temps pour pouvoir changer la fréquence de distribution de la nourriture. Nous avons donc choisi de faire une application web, comme cela permet d’avoir accès via différentes machines (téléphone, pc, etc.) et à différents endroits (à la maison ou hors de la maison).

Également, nous avons souhaité que l’utilisateur puisse consulter les moments où son chat est allé manger à son bol.

Voici donc notre interface web qui présente ces 2 aspects :

\*\*\*INSÉRER SCREENSHOT ICI\*\*\*

L’utilisateur peut donc incrémenter ou décrémenter de 30 minutes à la fois, en cliquant sur les boutons de chevrons, la fréquence de distribution de la nourriture. On bloque à 30 minutes minimum et 24 heures maximum. Ensuite il faut cliquer sur le bouton pour sauvegarder le changement.

Ensuite l’utilisateur voit un petit tableau de l’historique d’actionnement du servomoteur qu’il peut simplement consulter pour voir quand son chat a mangé.

Les informations consignées dans ce tableau proviennent de notre base de données, dans la table « ouverture\_distributeur », qui contient pour chaque entrée la date et le nombre d’ouvertures. On conserve une entrée par date, et à chaque ouverture le nombre va s’incrémenter dans la table.

On a aussi en BD une table « interval\_distributeur » qui contiendra toujours une seule entrée qui sera écrasée à chaque changement. On a un intervalle pour stocker la fréquence de distribution de la nourriture et une date de mise à jour pour savoir quand on a fixé cet intervalle.

Voir en Annexe 1 notre code SQL pour la création de nos tables en BD et l’insertion de données de test.

# Problèmes rencontrés

1. Problème : Avec l’ajout des requêtes HTTP dans le code arduino, nous manquions d’espace mémoire pour toutes nos variables donc on ne pouvait pas compiler.

Solution : On a déplacé une constante de la RAM à la PROGMEM pour laisser plus de place dans la RAM (stepSequence), car elle contient un tableau de tableaux ce qui semblait prendre pas mal de RAM.

1. Problème : On n’était pas en mesure de faire fonctionner le transfert de données via le ENC28J60 (pour la connexion à internet).

Solution : Après multiples essais, on a laissé tomber ce composant et on a fait notre connexion via le serial. Ça nécessite d’avoir le Arduino Uno branché dans un PC mais cela fonctionne bien.

1. Problème : On acait un problème d’affichage du nombre d’ouvertures dans notre tableau dans botre interface (mettons c’était initialement 1 et on en faisait une de plus, ça mettait 11, 111, etc. au lieu de 2, 3, 4...

Solution : Notre table SQL avait la colonne du nombre d’ouvertures en varchar au lieu de int donc ce n’était pas considéré comme un nombre. On a changé pour un int et ça fonctionne.

# Annexe 1 : Queries SQL pour la création des tables et l’insertion de données test

-- Table de configuration (1 seule entrée)

CREATE TABLE interval\_distributeur (

id\_int INT PRIMARY KEY,

interval\_ms\_int BIGINT NOT NULL,

updated\_at\_int DATETIME DEFAULT CURRENT\_TIMESTAMP ON UPDATE CURRENT\_TIMESTAMP

);

-- Table des logs d'ouvertures

CREATE TABLE ouverture\_distributeur (

id\_ouv INT AUTO\_INCREMENT PRIMARY KEY,

nb\_ouv VARCHAR(20),

date\_ouv DATE DEFAULT (CURRENT\_DATE),

UNIQUE (date\_ouv)

);

INSERT INTO interval\_distributeur (id\_int, interval\_ms\_int)

VALUES (1, 3600000);

INSERT INTO `object`.`ouverture\_distributeur`(`nb\_ouv`,`date\_ouv`)

VALUES (4, now());