



# RÉGULATION DE LA CONSOMMATION ÉLECTRIQUE **PILOTÉ** PAR LE SERVICE **EcoWATT**

N° DU PROJET : 40

## UE CODEVSI - RAPPORT TECHNIQUE

Date : 22 mai 2023

Version : 1.0

Formation/année : 2022-2023

Destinataires : Issam Rebaï, comité de pilotage de l'UE CODEVSI, étudiants FISE1A  
2023-2024 poursuivant le projet

Hugo Béchu, étudiant à IMT Atlantique en FISE1A

[hugo.bechu@imt-atlantique.net](mailto:hugo.bechu@imt-atlantique.net)

Clément Jolivet, étudiant à IMT Atlantique en FISE1A

[clement.jolivet@imt-atlantique.net](mailto:clement.jolivet@imt-atlantique.net)

Alexandre Epaillard, étudiant à IMT Atlantique en FISE1A

[alexandre.epaillard@imt-atlantique.net](mailto:alexandre.epaillard@imt-atlantique.net)

# RÉSUMÉ

Rédacteur : Alexandre Epaillard ▾ Relecteur : Clément Jolivet ▾

Dans un contexte de crise énergétique de plus en plus tendu, causant possiblement des risques de coupures d'électricité, ce projet vise à développer un système de gestion énergétique intelligent qui permettra aux utilisateurs de contrôler et de surveiller leur consommation électrique de manière efficace (en particulier pour ce qui est des équipements énergivores). Pour ce faire, il est nécessaire d'utiliser l'[API](#)<sup>1</sup> d'EcoWatt (développée par RTE) qui permet d'avoir des informations précises et actuelles sur la "météo énergétique" de la France. Grâce à cela, un système a pu être réalisé, permettant de réguler la consommation électrique des particuliers selon différents modes de régulation possible. Ainsi, dans un scénario où la production d'électricité est bien plus faible que la demande, si l'application est utilisée par un grand nombre de foyers, elle permettrait de limiter les risques de coupures d'électricité.

A travers ce rapport, les différentes étapes du développement du système sont détaillées, allant de l'analyse, de la conception du système et du développement du logiciel de contrôle, à l'intégration des différentes fonctionnalités. Des tests approfondis ont également été effectués pour garantir la fiabilité et la performance du système et sont présentés dans ce rapport. Ce dernier est destiné aux COPIL de l'UE CODEVSI, à l'encadrant Issam Rebaï, mais également à de futurs ingénieurs et aux futurs étudiants qui pourrait continuer ce projet sur l'année prochaine (année scolaire 2023-2024).

---

<sup>1</sup> Les textes soulignés de cette manière sont un lien pour les définitions présentes dans le glossaire

# SOMMAIRE

<b>I. Introduction.....</b>	<b>4</b>
<b>II. Gestion du projet.....</b>	<b>6</b>
Cahier des charges.....	6
Initiation aux outils de développement.....	7
<b>III. Conception du système.....</b>	<b>8</b>
Modélisation par flux.....	8
Conception des tests.....	11
<b>IV. Réalisation du système.....</b>	<b>12</b>
Programmation sur Node-RED.....	12
Implémentation de la récupération des données.....	12
Implémentation des fonctions principales.....	15
Modélisation matérielle.....	17
<b>V. Validation du système.....</b>	<b>18</b>
Application des tests.....	18
Interprétation des résultats.....	19
<b>VI. Conclusion.....</b>	<b>22</b>
<b>VII. Bibliographie.....</b>	<b>23</b>
<b>VIII. Glossaire.....</b>	<b>23</b>
<b>IX. Annexe.....</b>	<b>24</b>
Planning initial.....	24
Planning réalisé.....	27
Analyse comparative.....	30

## I. INTRODUCTION

Rédacteur : Clément Jolivet ▾ Relecteur : Alexandre Epaillard ▾

La consommation d'électricité des particuliers représente une part importante de la consommation totale en France puisqu'environ 40% de l'électricité produite est utilisée par ces derniers (le reste étant pour l'industrie, les entreprises, etc...). Ceci représente donc 40% de 460 TWh d'électricité consommé en 2020 <sup>[1]</sup><sup>2</sup>. Depuis 2022, la guerre en Ukraine a conduit la France dans une crise énergétique inédite entraînant une explosion des prix de l'électricité provenant des énergies fossiles. De plus, la production d'électricité en France a diminué, particulièrement dans le secteur du nucléaire avec près de 30% de baisse, au point qu'avec 279 TWh, le niveau de production de la filière nucléaire est le plus faible depuis 1988. Ceci est dû à la faible disponibilité du parc nucléaire français à cause des nombreuses maintenances. L'hydraulique, la seconde filière de production d'électricité, connaît aussi un recul de 20% par rapport aux dernières années, causée notamment par la sécheresse de l'été dernier. Elle est au plus bas depuis 1976. La France a donc été contrainte d'importer de l'électricité pour essayer de subvenir à la consommation habituelle des Français.

C'est dans ce contexte de crise énergétique qui sévit actuellement sur l'Europe que RTE (le Réseau de Transport d'Électricité en France) a eu l'idée de publier une API nommée EcoWatt. Elle permet de communiquer, à l'image d'un bulletin météorologique, le niveau de consommation électrique des Français estimé sur les trois jours à venir à l'aide d'un code couleur. En effet, dans l'idéal, pour éviter les problèmes de distribution, la production d'électricité doit être supérieure à la consommation du pays : ce scénario correspond au signal vert d'EcoWatt. Cependant, si la consommation prévue sur les trois jours tend à se rapprocher de la production, le signal deviendra orange (niveau tendu) et il est alors préférable d'économiser le plus possible l'électricité. Quant au cas où la situation devient très tendue (consommation supérieure à la production), le signal passe en rouge et il n'est plus possible de répondre à la demande totale d'électricité. Ainsi, cette API permet aux français de prendre les mesures nécessaires à leur niveau pour réduire leur consommation en électricité et ainsi éviter les risques de coupures de courant.

Ainsi, ce projet a pour objectif de développer une application capable d'exploiter les données fournies par l'API EcoWatt pour piloter automatiquement des équipements énergivores (radiateur électrique, ballon d'eau chaude, four, sèche-linge, etc.) dans le but d'ajuster de la meilleure manière sa consommation d'électricité. La problématique est alors : comment piloter automatiquement des équipements énergivores à partir des données de l'API EcoWatt ?

Pour répondre à cela à travers ce rapport, une procédure de développement ingénieur a été adoptée. Il sera donc présenté dans un premier temps la phase de gestion du produit, afin de bien comprendre les enjeux et bien situer ces derniers. Se pose alors le détail sur la question de la conception du système. La phase de conception s'ensuit d'une étape de

---

<sup>2</sup> Cet indice est un lien pour les références bibliographiques

réalisation, pour enfin achever la démarche ingénieure avec l'étape de validation du système. Les résultats de la démarche et du projet sont finalement synthétisés de manière globale dans la conclusion.

## II. GESTION DU PROJET

### CAHIER DES CHARGES

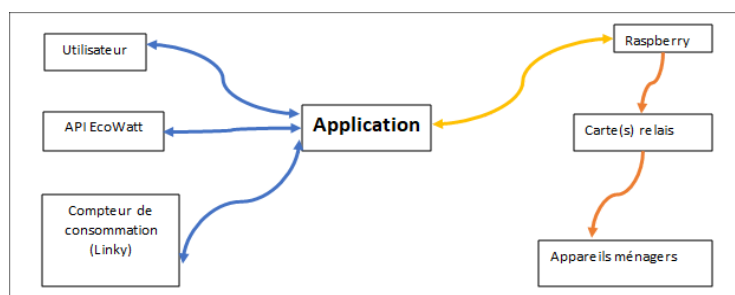
Rédacteur : Clément Jolivet ▾ Relecteur : Alexandre Epaillard ▾

#### PÉRIMÈTRE DU SYSTÈME

L'application développée devra pouvoir piloter les appareils électriques pour leur imposer leur mode de consommation. Pour cela, elle devra prendre en compte les données envoyées par EcoWatt comme le montre la Figure 1 avec les flèches bleues. L'application devra communiquer à distance avec des cartes relais (flèches jaunes sur la Figure 1) auxquelles seront branchées les appareils électroniques. L'application tournera en permanence. Elle devra différencier les modes de consommation pour chaque type d'appareils. Les modes à développer sont :

- Le mode régulation qui consiste à lisser la courbe de consommation en alternant le fonctionnement de plusieurs équipements. Ce mode s'applique par exemple pour piloter plusieurs radiateurs allumés en même temps.
- Le mode verrouillage qui s'applique quand l'utilisateur souhaite s'interdire l'usage d'un équipement quand le système électrique est tendu ou très tendu. C'est le cas par exemple du four ou du sèche-linge.
- Le mode délestage, qui consiste à débrancher automatiquement des équipements secondaires quand la consommation est élevée et dépasse un certain seuil. C'est le cas par exemple d'un ballon d'eau chaude ou de radiateurs.

L'application offrira une [IHM](#) de pilotage et de configuration pour permettre à l'utilisateur de changer le mode de fonctionnement. Elle ne devra pas mettre le mauvais mode de fonctionnement aux appareils. Elle ne devra pas non plus mettre en fonctionnement un appareil si la consommation électrique est dépassée et elle ne devra pas changer le mode de fonctionnement d'un appareil qui avait été configuré manuellement par l'utilisateur. L'utilisateur ne pourra pas directement intervenir sur les cartes relais, il devra impérativement passer par l'application développée étant donné qu'il faut prendre aussi en compte les informations envoyées par EcoWatt et [le compteur Linky](#).



#### Légende :

Flèches bleues : acteurs et informateurs décisionnels.

Flèche jaune : distributeur d'ordre.

Flèches orange : receveurs d'informations

**Figure 1 : le système dans son environnement**

## INITIATION AUX OUTILS DE DÉVELOPPEMENT

Rédacteur : Clément Jolivet ▾ Relecteur : Alexandre Epaillard ▾

Le développement de l'application nécessite des connaissances techniques sur des logiciels de programmation et des systèmes de partage de code qui ne sont pas maîtrisés au début du deuxième semestre de FISE A1. De plus, il est nécessaire d'assister à une formation afin de savoir comment utiliser un outil de modélisation en 3D. Il est donc important pour ce projet d'être initié aux trois outils de développement suivant :

- le logiciel de modélisation 3D SolidWorks
- le système de gestion de version décentralisée Git
- l'outil de planification Microsoft Project

SolidWorks est un logiciel de Conception Assistée par Ordinateur (CAO) largement utilisé dans le domaine de l'ingénierie et de la conception industrielle. Il permet de faire de la modélisation 3D, de la simulation, de la programmation de découpe ou d'impression 3D. Cela permet aux ingénieurs ou aux fabricants de créer des modèles virtuels de pièces, d'assemblages et de systèmes, facilitant ainsi le processus de conception. La formation permet alors de découvrir les bases de la conception et de la modélisation 3D avec la création de pièces, d'assemblages, et de mises en plan. De plus, elle initie aux méthodes de dimensionnement pour l'impression 3D. Cette formation apporte les connaissances suffisantes pour pouvoir à terme créer la boîte réceptionnant la carte relais et les prises de branchements. Cependant, il était ici préférable d'utiliser le logiciel de CAO 3D Fusion 360, certes moins fourni, moins professionnel, et moins complet que SolidWorks mais suffisant au projet et permettant un partage du modèle de conception virtuel entre les différents membres de l'équipe.

Git est un système de contrôle de version décentralisée largement utilisé dans le développement logiciel. Il permet aux ingénieurs en développement de gérer et de suivre les modifications apportées aux fichiers sources, facilitant ainsi la collaboration et la gestion du code. On peut ainsi conserver un historique des modifications et mieux gérer les conflits, contrairement à des systèmes de contribution simultanée comme Google Collabs qui peuvent écraser les versions des autres collaborateurs du code. Il est également pertinent d'utiliser GitHub afin de partager plus simplement les avancées du code entre les membres de l'équipe et avec l'encadrant.

Microsoft Project est un outil de gestion de tâches et de collaboration développé par Microsoft. Il permet aux équipes de créer des plans, d'organiser des tâches, de suivre les progrès et de collaborer de manière efficace et coordonnée. La formation introduit aux fonctionnalités de planification de Project comme la création de planning ( diagramme de Gantt) pour visualiser les tâches, les affecter à des membres de l'équipe, définir des dates d'échéance et suivre l'avancement grâce à des indicateurs visuels. Cela permet de mieux visualiser l'organisation du projet et de simplifier sa gestion, et aide également à rester

organisé et productif. Le détail des plannings prévisionnel et réel est donné en annexe du rapport.

L'analyse des besoins auxquels répond le système est maintenant bien identifiée, les formations aux outils de développement sont effectuées, ce qui permet alors de concevoir le système.

### III. CONCEPTION DU SYSTÈME

#### MODÉLISATION PAR FLUX

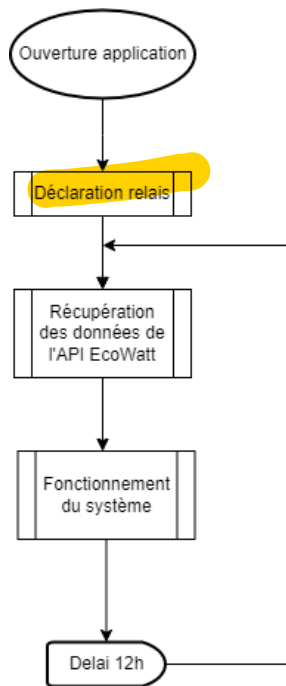
Rédacteur : Clément Jolivet ▾ Relecteur : Hugo Béchu ▾

#### CONCEPTION PAR FLUX

Afin de concevoir le système, il est intéressant de se pencher sur la modélisation par flux. C'est une approche méthodologique qui vise à structurer et à optimiser le processus de développement logiciel. Elle repose sur la création de flux d'activités interconnectées, où chaque étape représente une tâche spécifique dans le cycle de vie du développement. Cette approche favorise une gestion efficace de la programmation en divisant les tâches en modules distincts et en les reliant de manière séquentielle ou parallèle. Il était intéressant d'opter pour cette conception car cela correspondait à l'approche interactive et relationnelle nécessaire au bon fonctionnement du système. Les diagrammes de flux décrivant le fonctionnement du système ont alors été établis.

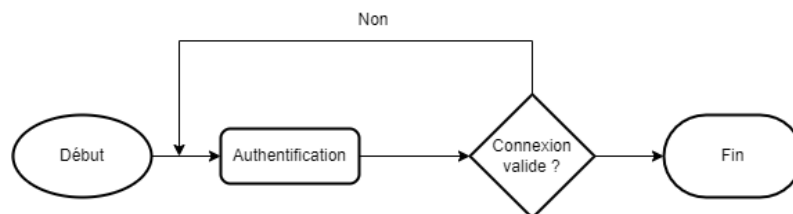
Le fonctionnement global du système décrit au Diagramme 1 s'accorde bien avec un fonctionnement par flux : l'utilisateur renseigne les paramètres et donne une consigne de fonctionnement, l'application récupère les données puis les analyse et les traite pour finalement renvoyer un résultat de fonctionnement. L'API fournissant des données sur 3 jours, il a été préférable d'opter pour une actualisation des données toutes les 12h une fois le système mis en marche.





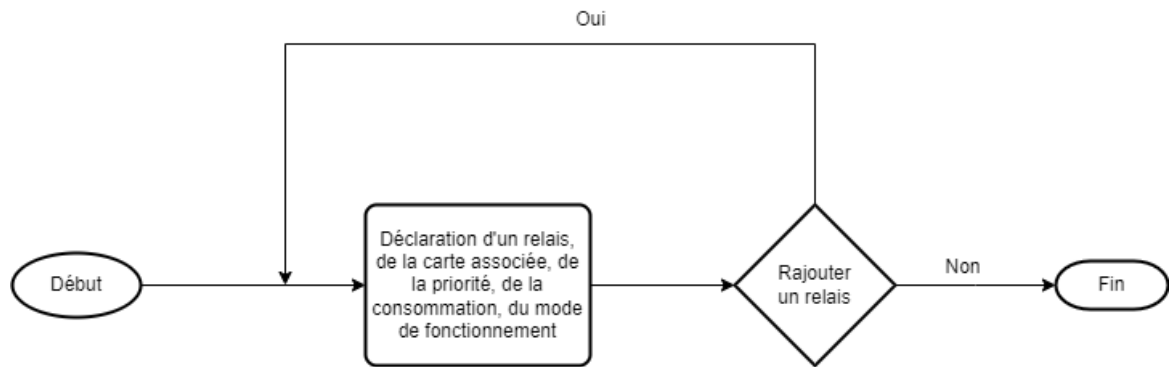
**Diagramme 1 : Fonctionnement général**

L'ouverture de l'application se joint d'une demande d'authentification (Diagramme 2) pour se connecter au système : il est en effet exigé qu'une seule personne puisse accéder à l'IHM. Une erreur de connexion mène à une demande d'authentification, et une connexion valide ouvre la section de déclaration de relais.



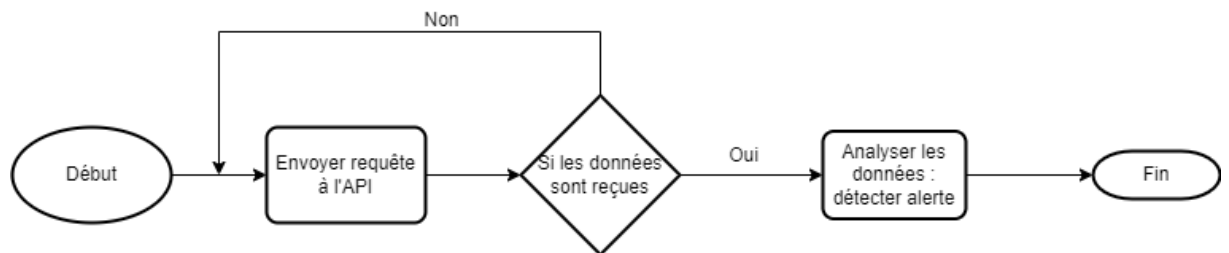
**Diagramme 2 : Procédé d'authentification**

Une fois connecté, l'utilisateur déclare un relais (Diagramme 3), la carte associée, le régime de priorité, la consommation de l'appareil branché, et le mode de fonctionnement. Il peut ajouter n relais. Une fois les relais déclarés, le système cherche à récupérer les données de l'API EcoWatt.



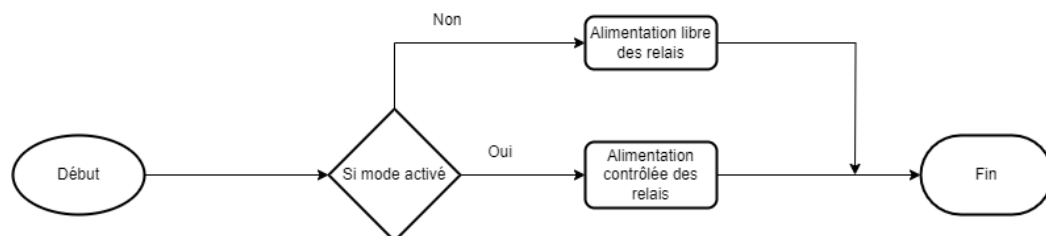
**Diagramme 3 : Déclaration des relais**

La récupération des données d'EcoWatt (Diagramme 4) constitue une majeure partie du développement logiciel. Elle se fait en envoyant une requête à l'API, puis le système vérifie la bonne réception des informations pour pouvoir ensuite les traiter et détecter une potentielle alerte.



**Diagramme 4 : Récupération des données**

Une fois les données récupérées et analysées, le système peut alors appliquer le mode de fonctionnement (Diagramme 5) adapté à la consigne de l'utilisateur et alimenter les relais en conséquence, ce qui en fait la seconde grande partie de développement du software. L'utilisateur peut contrôler directement les appareils si aucune alerte de consommation ne l'empêche avec l'activation d'un mode de fonctionnement. De plus, les modes s'actualisent lorsque l'utilisateur change une entrée (relais, priorité...).



**Diagramme 5 : Modes de fonctionnement**

Les diagrammes de flux permettent ainsi de mieux visualiser la procédure de fonctionnement du système et facilitent alors la programmation.

## CONCEPTION DES TESTS

Rédacteur : Alexandre Epaillard ▾ Relecteur : Clément Jolivet ▾

Des tests sur le produit doivent être effectués afin de valider chaque fonction et chaque contrainte listées précédemment dans le cahier des charges. En effet, à chaque fonction et à chaque contrainte sont associés des critères de validation. Il faut donc vérifier des critères temporels, d’affichage, de dimension, et de consommation électrique.

Afin de tester le système en fonction des données reçues par l’API EcoWatt, il n’est pas pertinent d’attendre une variation de la consommation en direct, celle-ci étant stable depuis plusieurs mois en France. Toutefois, lorsqu’on connecte Node-RED à l’API, il est proposé de recevoir les vraies données d’EcoWatt, ou bien de recevoir des données générées aléatoirement, résultant parfois en des scénarios de coupure d’électricité. Il est donc possible d’utiliser cette méthode pour étudier les tests de différents scénarios. Concernant les données potentiellement reçues par le compteur Linky, il faudra procéder de même en s’adaptant aux types de données envoyées par ce dernier ainsi qu’à leur format.

Les fonctionnalités étant caractérisées par un niveau temporel sont quant à elles soumises à des répétitions de chaque test afin de pouvoir établir une moyenne des résultats validant le niveau d’exigence lié à la fonctionnalité. En fonction du niveau de flexibilité, cette répétition de tests sert également d’indicateur de validation si un résultat limite tend à dépasser le niveau défini. L’établissement de ces tests peut être coûteux en temps, une vérification après chaque établissement d’une de ces fonctionnalités ou d’une étape de celle-ci fera certes augmenter le temps de réalisation mais permettra de détecter plus rapidement le moment où le niveau n’est plus respecté et fera ainsi réduire le temps de test final.

Pour les fonctionnalités qui concernent des déclarations de données, il faut envisager les cas de figures où les données sont saisies correctement et celles où elles ne le sont pas. Ces tests permettront de valider le fonctionnement du système dans les cas non nominaux et dans certains cas limites. Il est également important de vérifier que l’IHM affiche bien les données rentrées par l’utilisateur.

Pour ce qui est de la validation de l’activation des modes de fonctionnement, elle peut se faire via trois tests, un pour chaque mode de fonctionnement. En effet, il s’agit alors de lancer un mode de fonctionnement sur l’application et de vérifier qu’il est bien appliqué via la réaction des appareils connectés à la carte relais. Il faut ensuite vérifier que lorsque le mode est changé, le deuxième mode est bien actionné.

Les fonctionnalités liées à des visualisations de données sont quant à elles testables en vérifiant visuellement le bon fonctionnement de l’affichage dans les différents scénarios donnés.

## IV. RÉALISATION DU SYSTÈME

### PROGRAMMATION SUR NODE-RED

Rédacteur : Hugo Béchu ▾ Relecteur : Alexandre Epaillard ▾

Node-RED est une plateforme de développement visuel basée sur JavaScript qui permet la création d'applications et de flux de données. Elle offre une interface pour connecter des nœuds préfabriqués et créer des flux d'exécution. Node-RED permet aux développeurs de connecter facilement des API, des appareils IoT, des services en ligne et des bases de données, facilitant ainsi l'automatisation des tâches et la création d'applications interactives. C'est pour ces fonctionnalités efficaces et correspondant aux besoins du projet en termes de conception par flux pour le développement du système et de l'IHM que l'utilisation de cette plateforme de développement a été favorisée.

Tout d'abord, pour réaliser le code du projet qui pilotera automatiquement des équipements énergivores, il était logique de se reposer sur la modélisation par flux définie à la section précédente. Pour réaliser ce projet, le travail peut être séparé en deux parties : la récupération des données essentielles et l'implémentation des trois modes de fonctionnement des équipements (délestage, régulation, verrouillage). Pour simplifier le ~~programme~~ dans toute la suite le programme ne marchera que pour 4 relais et ne pourra pas fonctionner dynamiquement avec un paramètre nombre de relais que l'utilisateur aurait pu renseigner au début de l'installation du système. En effet, la plateforme Node-RED ne permet pas de prendre en compte un codage dynamique de première abord. Après avoir consulté plus amplement la documentation, il apparaît qu'une solution existe via un module de Node-RED nommé *uibuilder* [2] qui permet de créer des IHM dynamiquement. Son utilisation nécessite cependant des connaissances en HTML que nous n'avons pas et donc demande trop de temps pour être implémenté.

#### IMPLÉMENTATION DE LA RÉCUPÉRATION DES DONNÉES

Le premier objectif pour mener à bien le projet est de récupérer toutes les données nécessaires à l'implémentation des fonctions codant les trois modes de fonctionnement.

Pour cela, il faut définir ~~quelles seraient~~ les données nécessaires à l'implémentation de telles fonctions. Il est tout d'abord primordial d'avoir accès aux données d'EcoWatt pour pouvoir bien choisir le mode de fonctionnement adapté. Il faut aussi les données relatives à chaque relais, c'est-à-dire quel équipement est branché à quel relais, quel est son mode de fonctionnement privilégié, et quelle est la durée d'utilisation des équipements qui seront mis en mode régulation. En effet, il faut savoir à tout instant quel appareil est branché pour pouvoir adapter le pilotage en fonction des ces derniers.

Pour la récupération des données relatives aux relais, un module de Node-RED "Node-RED Dashboard" [3] a été utilisé, permettant de faire une IHM. Grâce à cette IHM,

l'utilisateur peut saisir toutes les données relatives aux équipements qu'il souhaite brancher et il est alors possible de les récupérer. L'IHM qui a été codée avec les noeuds du module Dashboard de Node-RED est montré Figure 2.

The screenshot shows a Node-RED dashboard with two main sections: 'Gestion' (Management) and 'Système' (System). The 'Gestion' section contains four relay entries, each with a dropdown menu for selecting a mode. The 'Système' section contains a toggle switch for 'Allumage système' (System ignition).

Relai	Équipement	Mode
Relai1	Radiateur	Régulation
Relai2	Radiateur	Régulation
Relai3	Four	Verrouillage
Relai4	Ballon d'eau chaude	Delestage

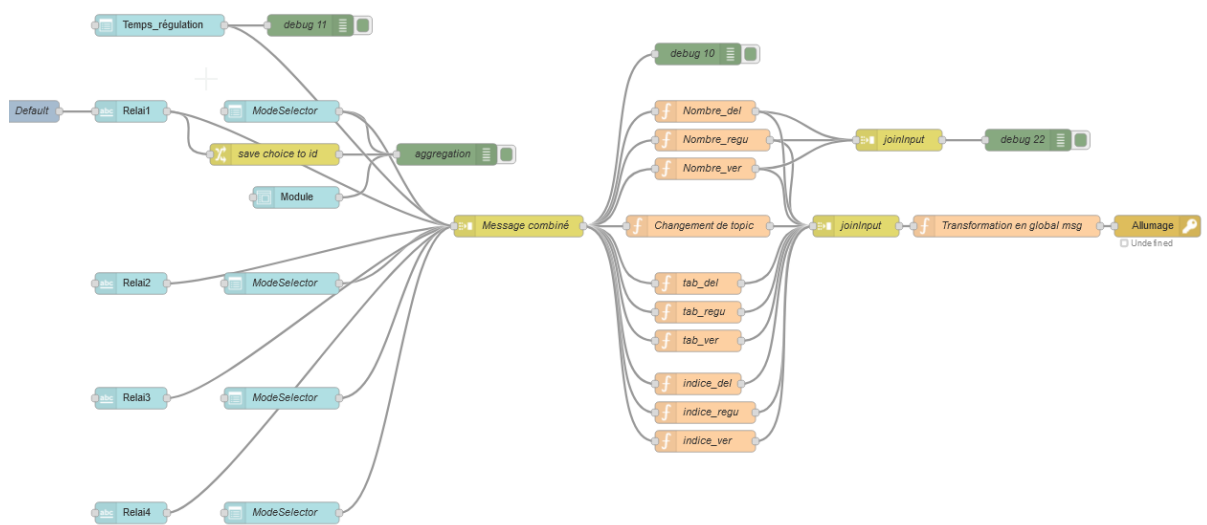
Temps\_régulation: 5 heures

Allumage système: ☐

**Figure 2 : IHM de déclaration des relais et des modes associés**

On peut voir qu'une IHM a été implémentée pour une carte de 4 relais. L'utilisateur y renseigne les équipements branchés aux différents relais puis indique le mode dans lequel il souhaite que ces équipements soient. Le champ Temps\_régulation permet à l'utilisateur de définir la durée d'utilisation des appareils branchés en mode régulation comme des radiateurs par exemple. Le switch "Allumage système" servira à allumer tout le système une fois toutes les données entrées.

Le codage d'une telle IHM sur Node-RED est présenté sur la Figure 3.

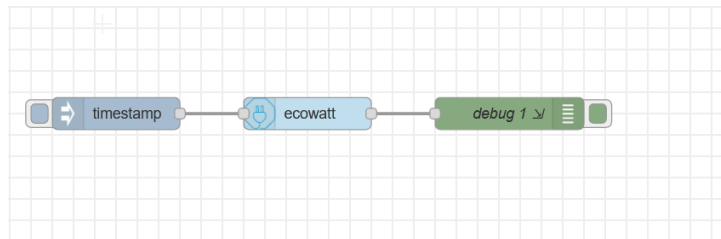


**Figure 3 : Récupération des données de l'IHM sur Node-RED**

Les nœuds en bleu clair à gauche sur l'image correspondent à chacune des valeurs saisies par l'utilisateur.

Néanmoins, une fois les données renseignées par l'utilisateur, il faut encore les rassembler dans un seul message, pour simplifier leur exploitation, ceci grâce au nœud "Message combiné". De plus, pour coder les fonctions à venir sur les modes de fonctionnement, il est nécessaire d'exploiter au préalable les données récupérées. Les fonctions en orange à droite du nœud "Message combiné" permettent de calculer le nombre d'équipement mis respectivement en mode délestage, régulation et verrouillage, ainsi que la liste des équipements par mode et l'indice du début de lecture de la liste. Une fois cela fait, il est possible de rassembler en un seul message les données et, avec la fonction "Transformation en global msg", de transmettre ce message à une variable global. Ceci permet d'accéder tout le temps aux différentes informations.

Les données relatives aux relais étant maintenant récupérées, il reste à récupérer les données d'EcoWatt pour savoir quand la consommation électrique est très élevée dans le pays. Pour cela, il est intéressant d'utiliser un nœud de Node-RED, préfabriqué par un utilisateur, nommé "EcoWatt" [4]. Un compte sur le site de RTE est ensuite créé et la clé d'authentification est renseignée dans le nœud. La Figure 4 montre une implémentation en Node-RED permettant de récupérer les données d'EcoWatt.

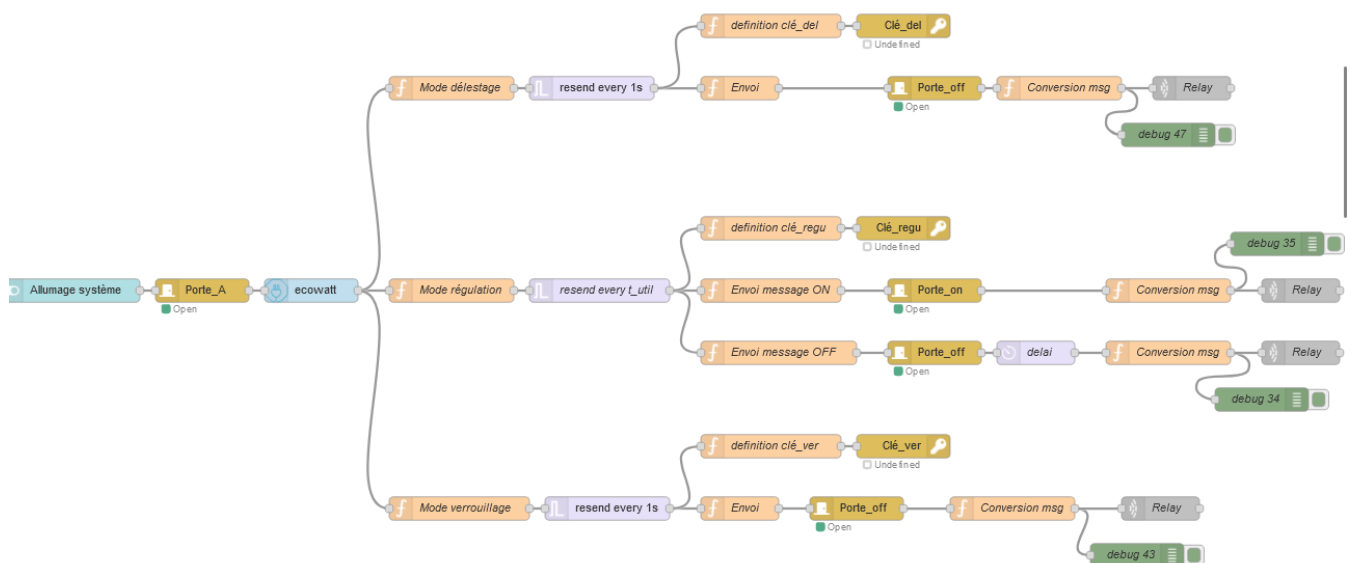


**Figure 4 : Récupération des données d'EcoWatt sur Node-RED**

## IMPLÉMENTATION DES FONCTIONS PRINCIPALES

Le second objectif est alors de programmer les fonctions principales du code qui définissent les différents modes de fonctionnement des équipements, c'est-à-dire qu'il faut coder des blocs de fonctions qui permettent d'implémenter le mode de régulation, le mode verrouillage et le mode délestage.

Après avoir bien défini les données en paramètre, les trois modes de fonctionnement ont été implémentés. La partie du code qui génère les trois modes de fonctionnement est sur la Figure 5.



**Figure 5 : Implémentation des 3 modes de fonctionnement sur Node-RED**

On peut tout d'abord voir à gauche de la Figure 5 un nœud switch "Allumage système" qui permet de commander l'allumage du système. Le nœud "Porte\_A" permet de laisser passer le message d'allumage si et seulement si les données relatives aux relais ont été bien récupérées (la bonne récupération des données sera expliquée dans la sous-partie suivante). Ensuite, le nœud "EcoWatt" permet de récupérer les données fournies par l'API

EcoWatt qui sont transmises alors aux 3 branches du code implémentant les modes de fonctionnement (la branche du haut correspondant au mode délestage, celle du milieu au mode régulation et celle du bas au mode verrouillage).

La branche du milieu implémente quant à elle le mode régulation. Tout d'abord, le temps d'utilisation de chaque équipement mis en mode régulation par l'utilisateur est calculé grâce à la variable global regroupant toutes les données et donnant entre autres aussi le nombre d'équipements mis en mode régulation et le temps d'utilisation global pour ces équipements. Ce temps d'utilisation de chaque équipement correspond ainsi à la division entre le temps d'utilisation global par le nombre d'équipement mis en mode régulation, il est noté  $t_{util}$ . Une fois ce temps calculé avec le noeud fonction "Mode régulation" codé en javascript, des messages sont envoyés à intervalle régulier toutes les  $t_{util}$  secondes avec le noeud trigger "reset every  $t_{util}$ ". Par la suite, un message est envoyé, toutes les  $t_{util}$  secondes, d'allumage des relais avec la fonction "Envoie message ON" pour allumer les relais un par un. Le message éteignant les relais un par un est envoyé avec un délai de  $t_{util}$  secondes pour laisser  $t_{util}$  secondes de fonctionnement au relais. Les différents messages sont convertis par la fonction "Conversion msg" pour que la carte relais puisse les comprendre (la carte a été programmé d'une façon à allumer/ éteindre les relais si elle reçoit un message dans un certain format) et ils sont envoyés à la carte via le noeud "relay" qui utilise un requête TCP. Pour éviter d'envoyer des messages d'erreur dû au fait que le cycle de régulation a été fini, deux portes "Porte\_on" et "Porte\_off" bloquent la transmission des messages aux noeuds suivants si la clé "Clé\_régu" est désactivée. La clé est désactivée quand la liste des équipements en mode régulation a été parcourue entièrement (c'est-à-dire quand indice\_regu est égale à la taille de la liste -1). Il est possible d'utiliser de tels noeuds, porte et clé, en important ces noeuds créés par un utilisateur de Node-RED [\[5\]](#).

La branche du haut implémente le mode délestage. Il faut regarder tout d'abord si les données d'EcoWatt avertissent d'une consommation élevée en France puis, sur le même principe que la branche codant le mode régulation, des messages sont envoyés à la carte relais pour dire d'éteindre les équipements qui sont mis en mode délestage. Il faut parcourir la liste d'équipements mis en mode délestage et quand elle est parcourue entièrement, il faut couper la transmission des messages grâce à un noeud porte comme expliqué au paragraphe précédent.

La branche du bas implémente le mode verrouillage. De même que pour la branche implémentant le mode délestage, il faut regarder si les données d'EcoWatt avertissent d'une tension puis envoyer des messages à la carte relais pour dire d'éteindre les équipements qui sont mis en mode verrouillage en s'arrêtant quand la liste des équipements a été parcourue entièrement grâce à un noeud porte.

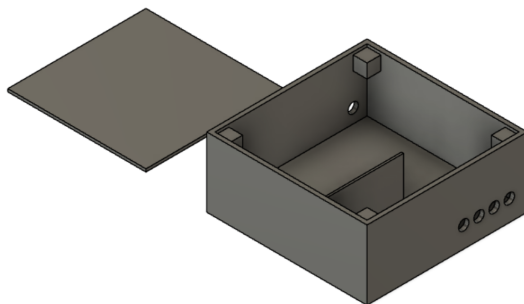
La phase de développement des fonctions implémentant la récupération des données et des fonctions principales du système étant ainsi détaillée, la partie *software* nécessite alors un support matériel par lequel elle pourra activer les modes de fonctionnement voulus et l'alimentation des relais correspondant.



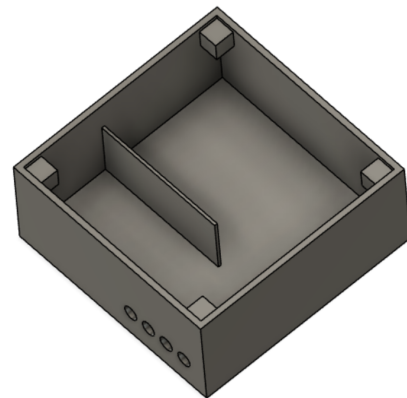
## MODÉLISATION MATÉRIELLE

Rédacteur : Alexandre Epaillard ▾ Relecteur : Clément Jolivet ▾

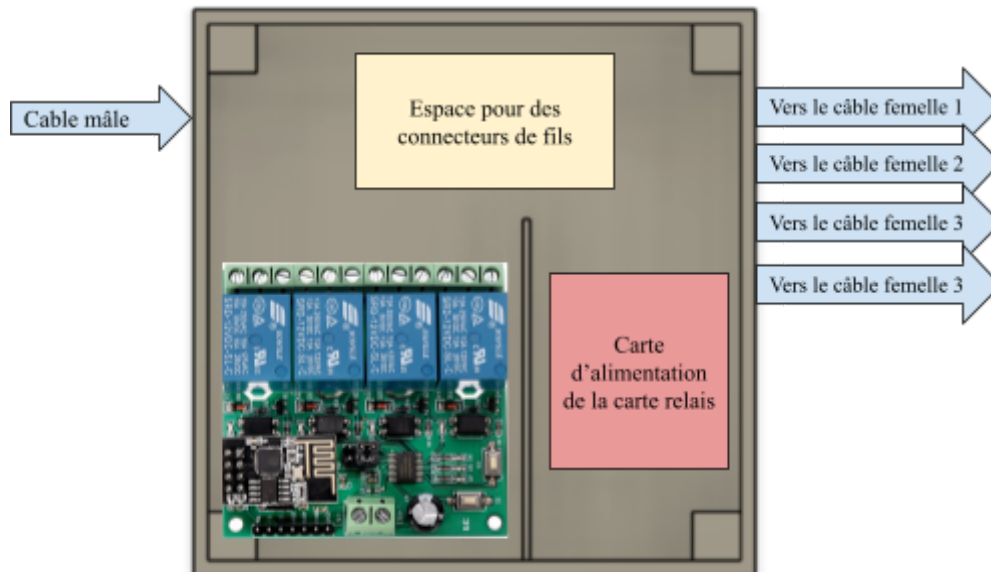
Le boîtier est dimensionné afin d'accueillir la carte relais, sa carte d'alimentation, des connecteurs entre la prise mâle et les différentes prises femelles. Les dimensions du boîtier sont de 106x106x30 mm. La carte relais mesurant 63x60x20 mm, elle s'insère alors dans le boîtier, plus particulièrement dans le compartiment de gauche (voir Schéma 1). Le compartiment de droite sert à accueillir la carte d'alimentation de la carte relais, et l'espace supérieur permet de faire la connectique entre le câble de la prise mâle branché au secteur et la carte d'alimentation, et entre les relais et les quatre câbles menant aux prises femelles sur lesquelles sont branchées les appareils. La connexion entre la carte relais et son alimentation se fait par dessus l'encoche séparatrice (Figure 5 et Figure 6). Les câbles utilisés sont de type H07VR et nécessitent alors l'utilisation de connecteurs afin de les relier aux différentes cartes. La hauteur du boîtier permet d'accueillir la carte Raspberry, et donc de finalement contenir tout le système. Le matériau utilisé est du PLA car il est possible de produire ce boîtier spécifique par impression 3D en PLA contrairement à du PVC. Il est cependant moins résistant que le PVC.



**Figure 6 : rendu 3D du boîtier**



**Figure 7 : rendu 3D du boîtier**



**Schéma 1 : organisation du boîtier**

L'implémentation des différentes fonctions, la réalisation du code nécessaire à la mise en œuvre du projet et la modélisation matérielle finies, il convient de définir différents tests et de les mettre en place pour valider ou non le système dans son ensemble par rapport aux critères définis dans le cahier des charges.

## V. VALIDATION DU SYSTÈME

### APPLICATION DES TESTS

Rédacteur : Hugo Béchu ▾ Relecteur : Clément Jolivet ▾

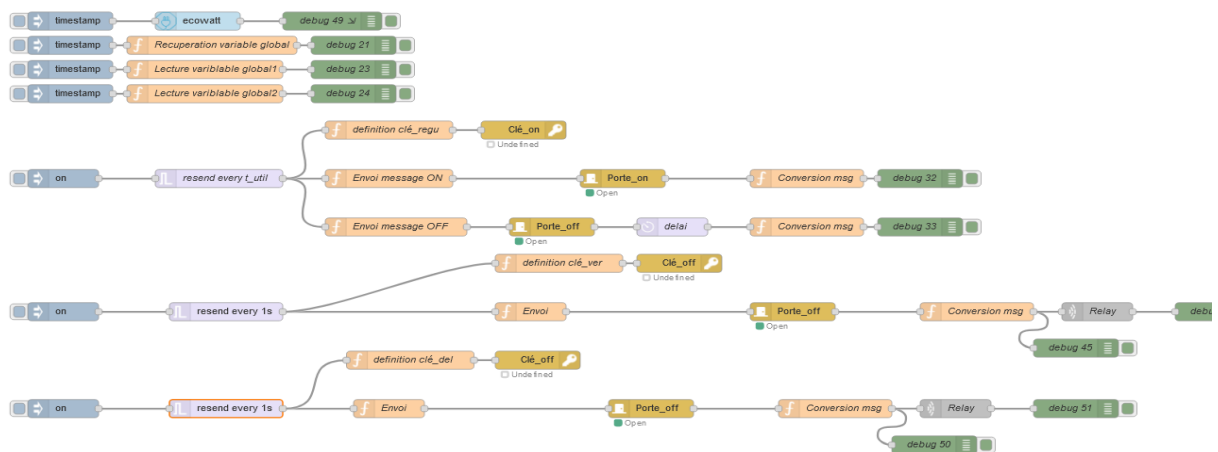
Plusieurs tests ont alors été réalisés afin de valider les fonctionnalités et les contraintes précédemment listées dans le cahier des charges. Certains de ces tests peuvent directement se réaliser via l'environnement Node-RED en l'utilisant à bon escient. En effet, les tests ont été codés au préalable afin de les implémenter ensuite sur Node-RED. Une fois le code de test écrit, un nouveau flow sur Node-RED a été créé afin d'y placer seulement les nœuds qui étaient nécessaires pour le test. Pour implémenter alors le test, il faut l'écrire dans un nœud "Function" et le relier aux autres nœuds, ainsi qu'à un nœud "Debug", permettant quant à lui d'afficher les résultats du test.

Il est donc possible de mettre un certain nombre de tests en place à l'aide de cette méthode, comme par exemple le test des données reçues par l'API EcoWatt. Lors de ce test, on veut vérifier si les données reçues par l'application via l'API EcoWatt sont bien du format [JSON](#) (comme indiqué sur le guide d'utilisation de l'API disponible sur le site EcoWatt) avec la date de génération des données, le jour auquel réfèrent les données, la valeur de ce jour

(comprise entre 1 et 3) ainsi que la valeur de chaque heure de ce jour, et ce pour 3 jours consécutifs. Ce test s'effectue sous Node-RED avec le premier flux (en partant du haut) sur la Figure 8.

Il est ensuite nécessaire de vérifier la bonne récupération des données saisies par l'utilisateur sur l'IHM. Pour cela, les flux 2, 3 et 4 (en partant toujours du haut) de la Figure 8 sont utilisés. La vérification montre que les données ont bien été récupérées et qu'elles s'actualisent au moindre changement.

Il reste ensuite à vérifier que les données sont correctement traitées par l'application en fonction des différents modes auxquels les équipements sont définis, en regardant si les résultats retournés sont cohérents avec les données envoyées, c'est-à-dire en s'assurant que les bons relais ont bien été allumés/éteints aux bons instants. Il est donc nécessaire de vérifier que l'application renvoie des erreurs ou des messages d'avertissement lorsque des données incorrectes sont envoyées. Pour maximiser la pertinence de ces tests, ces étapes sont répétées pour différentes combinaisons de données afin de vérifier que l'application gère correctement tous les cas de figure.



**Figure 8 : Implémentation des tests sous Node-RED**

## INTERPRÉTATION DES RÉSULTATS

Rédacteur : Hugo Béchu Relecteur : Alexandre Epailard

Une fois les différents tests définis, il convient de les appliquer à notre code. Après avoir effectué les différents tests présentés dans la partie précédente, les résultats apparaissent en conformité avec les attentes du cahier des charges. Détaillons les résultats des tests les plus essentiels.

Pour le 1er test sur la conformité de la réception des données d'EcoWatt, le test a donné le résultat présenté sur la Figure 9.

```

22/05/2023 21:51:59 node: debug 49
msg.payload: Object
▼ object
  ▼ signals: array[4]
    ▼ 0: object
      GenerationFichier:
        "2023-05-21T22:00:00+02:00"
      jour: "2023-05-22T00:00:00+02:00"
      dvalue: 1
      message: "Pas d'alerte."
      values: array[24]
    ▼ 1: object
      GenerationFichier:
        "2023-05-21T22:00:00+02:00"
      jour: "2023-05-23T00:00:00+02:00"
      dvalue: 1
      message: "Pas d'alerte."
      values: array[24]
    ▼ 2: object
      GenerationFichier:
        "2023-05-21T22:00:00+02:00"
      jour: "2023-05-24T00:00:00+02:00"
      dvalue: 1
      message: "Pas d'alerte."
      values: array[24]
    ▼ 3: object

```

**Figure 9 : Résultat du test d'EcoWatt sous Node-RED**

On remarque bien que le message renvoyé par le débogueur correspond au format de données émis par l'API EcoWatt et le message est conforme dans le fond : la date de génération des données, le jour auquel réfèrent les données, la valeur de ce jour (comprise entre 1 et 3) ainsi que la valeur de chaque heure de ce jour, et ce pour 3 jours consécutifs.

Le test de récupération des données de l'IHM configurée comme pour la Figure 3 (partie IV) donne le résultat de la Figure 10.

```

22/05/2023 23:28:46 node: debug 21
msg.payload: Object
▼ object
  nbr_del: 1
  nbr_regu: 2
  nbr_ver: 1
  donnees_IHM: object
    relayNumber1: "Radiateur"
    relayNumber2: "Radiateur"
    relayNumber3: "Four"
    relayNumber4: "Balon d'eau chaude"
    mode1: "regu"
    mode2: "regu"
    mode3: "ver"
    mode4: "del"
    temps: 5
  ▼ tab_del: array[1]
    0: 4
  ▼ tab_regu: array[2]
    0: 1
    1: 2
  ▼ tab_ver: array[1]
    0: 3
  indice_del: 0
  indice_regu: 0
  indice_ver: 0

```

**Figure 10 : Résultat du test des données de l'IHM sous Node-RED**

Le test est ainsi concluant puisque le message informe bien des données réellement saisies pour l'IHM configurée comme pour la figure 3.

Pour finir, le test du bon fonctionnement des modes donne par exemple pour le mode de régulation avec une IHM configurée de la même manière que le test précédent et un temps d'utilisation de chaque équipement de 5 secondes (Figure 11) :

```
22/05/2023 23:30:31 node: debug 32
1 : msg : Object
  { _msgid: "8c4e21d24b7ef0df", payload:
    buffer[4], topic: 1, delay: 5000,
    value: 2 ... }

22/05/2023 23:30:36 node: debug 33
1 : msg : Object
  { _msgid: "8c4e21d24b7ef0df", payload:
    buffer[4], topic: 1, delay: 5000,
    value: 1 ... }

22/05/2023 23:30:36 node: debug 32
2 : msg : Object
  { _msgid: "8c4e21d24b7ef0df", payload:
    buffer[4], topic: 2, delay: 5000,
    value: 3 ... }

22/05/2023 23:30:41 node: debug 33
2 : msg : Object
  { _msgid: "8c4e21d24b7ef0df", payload:
    buffer[4], topic: 2, delay: 5000,
    value: 2 ... }
```

**Figure 11 : Résultat du test du mode régulation sous Node-RED**

Les résultats indiquent que à 23:30:31 le premier relais est allumé puis 5 secondes plus tard il est éteint et c'est le relais 2 qui est allumé puis éteint 5 secondes plus tard à 23:30:41. Ceci correspond donc bien à un fonctionnement du mode régulation avec un temps d'utilisation de 5 secondes et où seuls les relais 1 et 2 ont été définis en mode régulation.

Néanmoins, concernant la partie matérielle, le boîtier du système n'ayant pas été réalisé, certains tests n'ont pas pu être effectués. En effet, comme les connections entre les différents composants n'ont pas pu être faites, les contraintes de la partie hardware du cahier des charges "Respecter les caractéristiques des composants", "Avoir un câblage adapté aux puissances des équipements" et "Éviter les dangers d'électrocution lors des connections à la carte relais" n'ont pas pu être vérifiées.

## VI. CONCLUSION

Rédacteur : Hugo Béchu ▾ Relecteur : Clément Jolivet ▾

Ce projet a donc pour but de développer une application capable d'exploiter les données fournies par l'API EcoWatt afin de piloter automatiquement des équipements énergivores. Le présent rapport permet de voir comment ce projet a été mené à travers les étapes de conception, de réalisation et de validation. La modélisation par flux a permis d'implémenter le système grâce à la plateforme Node-RED, le tout en vérifiant le bon comportement du système via plusieurs tests conceptionnés en fonction des cas nominaux et limites d'utilisation. Les résultats obtenus pour la partie ~~logiciel~~ montrent un comportement adéquat du système quant aux fonctionnalités déduites de l'analyse du besoin et aux critères associés.

L'application présente tout de même de nombreuses perspectives d'avenir que ce soit sur le court, moyen ou long terme. Étant donné que le programme a seulement été réalisé en ne prenant en compte que 4 relais, il convient, pour une utilisation plus intéressante et plus modulable de l'application, de réaliser une IHM dynamique qui pourra prendre en compte le nombre de relais et le nombre de cartes sur lesquels l'application sera implantée. Ceci pourra être implémenté à court terme à l'aide de *uibuilder*, comme expliqué dans le paragraphe sur la programmation sur Node-Red. Par ailleurs, la réalisation de la partie matérielle avec le câblage des différents composants est une perspective proche puisque la modélisation a été effectuée.

De plus, l'application pourra aussi gérer automatiquement la consommation des différents appareils électroniques en fonction de la consommation d'électricité de l'utilisateur grâce au compteur Linky. Cette perspective correspond à un moyen terme puisque ce n'est pas une partie essentielle du projet.

A très long terme, l'application pourra potentiellement être utilisée dans de nombreux logements par des utilisateurs qui souhaitent diminuer leur consommation électrique et mieux s'adapter aux tensions sur la consommation.

## VII. BIBLIOGRAPHIE

- [1] RTE-France, Bilan électrique 2020. Disponible sur : <https://bilan-electrique-2020.rte-france.com/> (consulté le 01/03/2023)
- [2] NODE-RED.org, node-red-contrib-uibuilder [en ligne]. Disponible sur : <https://flows.nodered.org/node/node-red-contrib-uibuilder> (consulté le 15/05/2023)
- [3] NODE-RED.org, node-red-dashboard [en ligne]. Disponible sur : <https://flows.nodered.org/node/node-red-dashboard> (consulté le 20/03/2023)
- [4] NODE-RED.org, node-red-dashboard [en ligne]. Disponible sur : <https://flows.nodered.org/node/@vital91/node-red-contrib-ecowatt> (consulté le 27/03/2023)
- [5] NODE-RED.org, node-red-dashboard [en ligne]. Disponible sur : <https://flows.nodered.org/node/node-red-contrib-message-gate> (consulté le 15/05/2023)

## VIII. GLOSSAIRE

API : interface de programmation d'application. Une API est une interface logicielle qui permet de « connecter » un logiciel ou un service à un autre logiciel ou service afin d'échanger des données et des fonctionnalités.

Carte IoT Relai : Carte à laquelle on branche les différents appareils que l'on souhaite contrôler. Cette carte est connectée via wifi à une carte Raspberry.

Compteur Linky : Un compteur Linky est un compteur électrique intelligent qui mesure la consommation d'électricité et permet une communication bidirectionnelle des données avec le fournisseur d'énergie.

IHM : une interface homme-machine fait référence à un tableau de bord qui permet à un utilisateur de communiquer avec une machine, un programme informatique ou un système.

JSON : format de données textuel dérivé de la notation des objets du langage JavaScript.

Raspberry : Le Raspberry Pi est un nano-ordinateur monocarte à processeur ARM de la taille d'une carte de crédit.

RTE : Réseau de Transport d'Électricité, c'est une société anonyme, filiale à 100% d'EDF. Sa neutralité à l'égard de tous les producteurs d'électricité est garantie par ses statuts.

## IX. ANNEXE

### PLANNING INITIAL

N°	Mod	Nom de la tâche	Durée	Début	Fin	Prédécesseurs	Noms ressources	27	Février 2023	01	06	11	16	21
1		<b>Analyse du besoin</b>	<b>5,6 semaines</b>	<b>Mer 01/02/23</b>	<b>Ven 10/03/23</b>									
2		Réunion obligatoire encadrant/élèves	0 jour	Mer 01/02/23	Mer 01/02/23		Ressources humaines		01/02					
3		Découverte du projet + recherche sur le projet	5 jours	Jeu 02/02/23	Mer 08/02/23	2	Fiche de présentation du projet + internet							
4		Réunion encadrant/élèves	0 jour	Jeu 09/02/23	Jeu 09/02/23	3	Ressources humaines							
5		Écriture du cahier des charges	20 jours	Ven 10/02/23	Jeu 09/03/23	4;7;10;13;6	Ressources humaines							
6		Cours et TD sur l'analyse du besoin du client	1 jour	Mer 15/02/23	Mer 15/02/23		Ressources humaines							
7		Réunion encadrant/élèves	0 jour	Jeu 16/02/23	Jeu 16/02/23		Ressources humaines							
8		Formation de gestion de délais	1 jour	Mer 22/02/23	Mer 22/02/23		Logiciel MS project							
9		Réalisation du planning	11 jours	Jeu 23/02/23	Jeu 09/03/23	8	Logiciel MS project							
10		Réunion encadrant/élèves	0 jour	Jeu 23/02/23	Jeu 23/02/23		Ressources humaines							
11		Formation Git	1 jour	Mer 01/03/23	Mer 01/03/23		Logiciel Git							
12		Formation SolidWorks	1 jour	Mer 01/03/23	Mer 01/03/23		Logiciel Solidworks							
13		Réunion obligatoire encadrant/élèves	0 jour	Jeu 09/03/23	Jeu 09/03/23		Ressources humaines							
14		Dépôt CdC + planning	0 jour	Ven 10/03/23	Ven 10/03/23	9;5	Ressources humaines							
15		<b>Conception/réalisation/validation</b>	<b>61 jours?</b>	<b>Ven 10/03/23</b>	<b>Jeu 01/06/23</b>									
16		<b>Réflexions techniques</b>	<b>61 jours</b>	<b>Ven 10/03/23</b>	<b>Jeu 01/06/23</b>									
17		Recherche des technologies à employer	6 jours	Ven 10/03/23	Ven 17/03/23	11;12	Internet + Ressources humaines							
18		Vacances	7 jours	Sam 25/03/23	Dim 02/04/23		Ressources humaines							

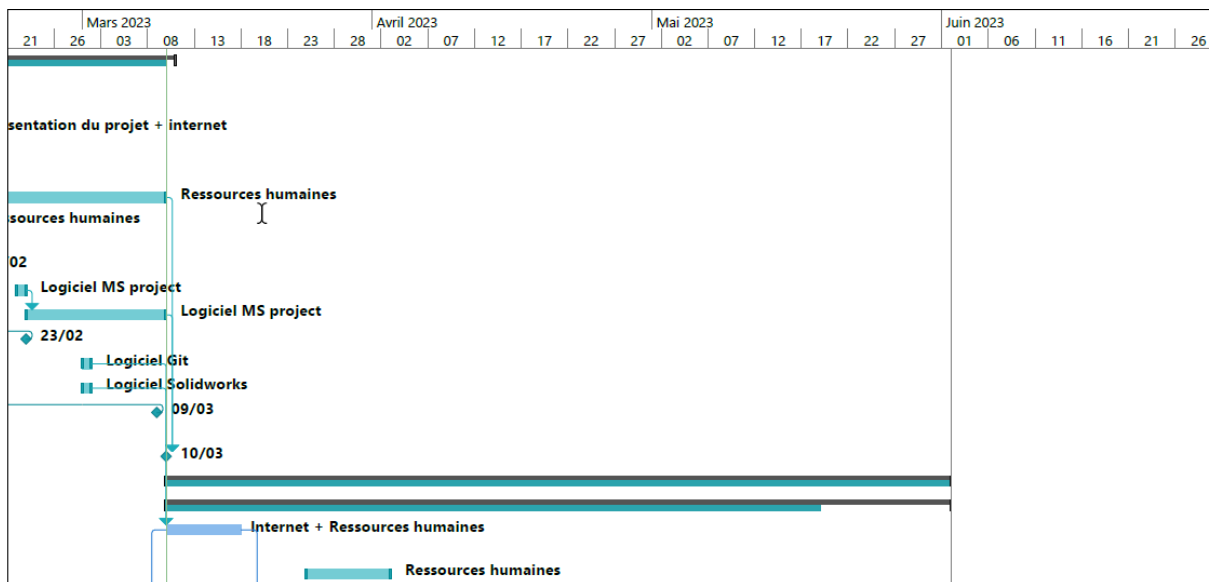
  

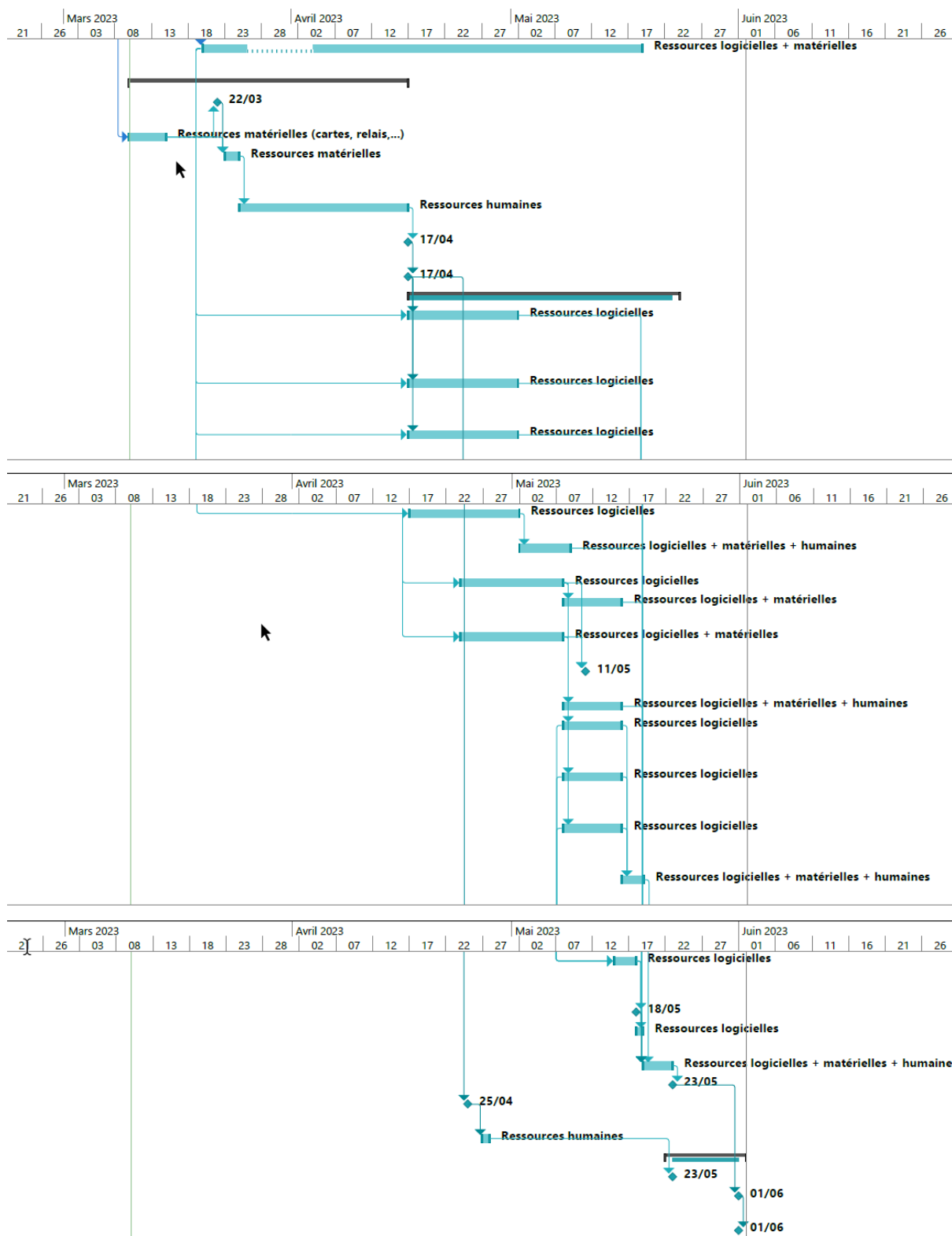
N°	Mod	Nom de la tâche	Durée	Début	Fin	Prédécesseurs	Noms ressources	27	Février 2023	01	06	11	16	21
19		Formation aux technologies trouvées	39 jours	Lun 20/03/23	Jeu 18/05/23	17	Ressources logicielles + matérielles							
20		<b>Conception</b>	<b>27 jours</b>	<b>Ven 10/03/23</b>	<b>Lun 17/04/23</b>									
21		Réunion obligatoire encadrant/élèves	0 jour	Mer 22/03/23	Mer 22/03/23	22	Ressources humaines							
22		Choix du matériel	3 jours	Ven 10/03/23	Mar 14/03/23	17DD	Ressources matérielles (c							
23		Identifier les différentes interactions entre les différents composants	2 jours	Jeu 23/03/23	Ven 24/03/23	22;21	Ressources matérielles							
24		Réalisation du modèle de l'application	17 jours	Sam 25/03/23	Dim 16/04/23	23	Ressources humaines							
25		Réunion obligatoire encadrant/élèves	0 jour	Lun 17/04/23	Lun 17/04/23	24	Ressources humaines							
26		Fin de la phase de conception	0 jour	Lun 17/04/23	Lun 17/04/23	25	Ressources humaines							
27		<b>Réalisation et tests</b>	<b>27 jours</b>	<b>Lun 17/04/23</b>	<b>Mar 23/05/23</b>									
28		Réalisation du code permettant d'appliquer un mode de fonctionnement à un équipement	11 jours	Lun 17/04/23	Lun 01/05/23	19DD;26	Ressources logicielles							
29		Réalisation du code permettant de spécifier le mode de fonctionnement	11 jours	Lun 17/04/23	Lun 01/05/23	19DD;26	Ressources logicielles							
30		Réalisation du code pour piloter l'équipement	11 jours	Lun 17/04/23	Lun 01/05/23	19DD;26	Ressources logicielles							



N°	Mod Tâche	Nom de la tâche	Durée	Début	Fin	Prédécesseurs	Noms ressources	27	Février 2023	01	06	11	16	21
31		Réalisation du code permettant de traiter les	11 jours	Lun 17/04/23	Lun 01/05/23	19DD	Ressources logicielles							
32		Test de la bonne récupération des données	5 jours	Mar 02/05/23	Lun 08/05/23	31	Ressources logicielles + matérielles + humaines							
33		Développement de l'IHM	11 jours	Lun 24/04/23	Dim 07/05/23	31DD	Ressources logicielles							
34		Test de fonctionnement de l'IHM	6 jours	Lun 08/05/23	Lun 15/05/23	33	Ressources logicielles + matérielles							
35		Elablisement des connexions entre les	11 jours	Lun 24/04/23	Dim 07/05/23	31DD	Ressources logicielles + matérielles							
36		Réunion obligatoire encadrant/élèves	0 jour	Jeu 11/05/23	Jeu 11/05/23	33;35	Ressources humaines							
37		Test des connexions	6 jours	Lun 08/05/23	Lun 15/05/23	335	Ressources logicielles + m							
38		Configuration de l'IHM pour permettre la déclaration de la configuration des	6 jours	Lun 08/05/23	Lun 15/05/23	33	Ressources logicielles							
39		Configuration de l'IHM pour permettre la déclaration de les priorités sur le	6 jours	Lun 08/05/23	Lun 15/05/23	33	Ressources logicielles							
40		Configuration de l'IHM pour permettre la déclaration du nombre de cartes et de	6 jours	Lun 08/05/23	Lun 15/05/23	33	Ressources logicielles							
41		Test de l'IHM pour les différentes déclarations	3 jours	Mar 16/05/23	Jeu 18/05/23	38;39;40	Ressources logicielles + matérielles + humaines							

N°	Mod Tâche	Nom de la tâche	Durée	Début	Fin	Prédécesseurs	Noms ressources	27	Février 2023	01	06	11	16	21
42		Réalisation du code permettant de traiter les données du compteur Linky	3 jours	Lun 15/05/23	Mer 17/05/23	40DD;39DD;38	Ressources logicielles							
43		Fin de la phase de réalisation	0 jour	Jeu 18/05/23	Jeu 18/05/23	42	Ressources humaines							
44		Test de la bonne récupération des données	1 jour	Jeu 18/05/23	Jeu 18/05/23	42	Ressources logicielles							
45		Test global de l'application	2 jours	Ven 19/05/23	Lun 22/05/23	41;43;37;34;32	Ressources logicielles + m							
46		Fin de la phase de test	0 jour	Mar 23/05/23	Mar 23/05/23	45	Ressources humaines							
47		QCM Moodle (atelier aide rédaction support)	0 jour	Mar 25/04/23	Mar 25/04/23	26	Ressources humaines							
48		Atelier de rédaction du rapport	1 jour	Jeu 27/04/23	Jeu 27/04/23	47	Ressources humaines							
49		Communication écrite/restitutio	9 jours?	Lun 22/05/23	Jeu 01/06/23									
50		Dépôt du rapport	0 jour	Mar 23/05/23	Mar 23/05/23	48	Ressources humaines							
51		Réunion obligatoire encadrant/élèves	0 jour?	Jeu 01/06/23	Jeu 01/06/23	46	Ressources humaines							
52		Restitution finale (code, proto.	0 jour	Jeu 01/06/23	Jeu 01/06/23	51	Ressources humaines							
















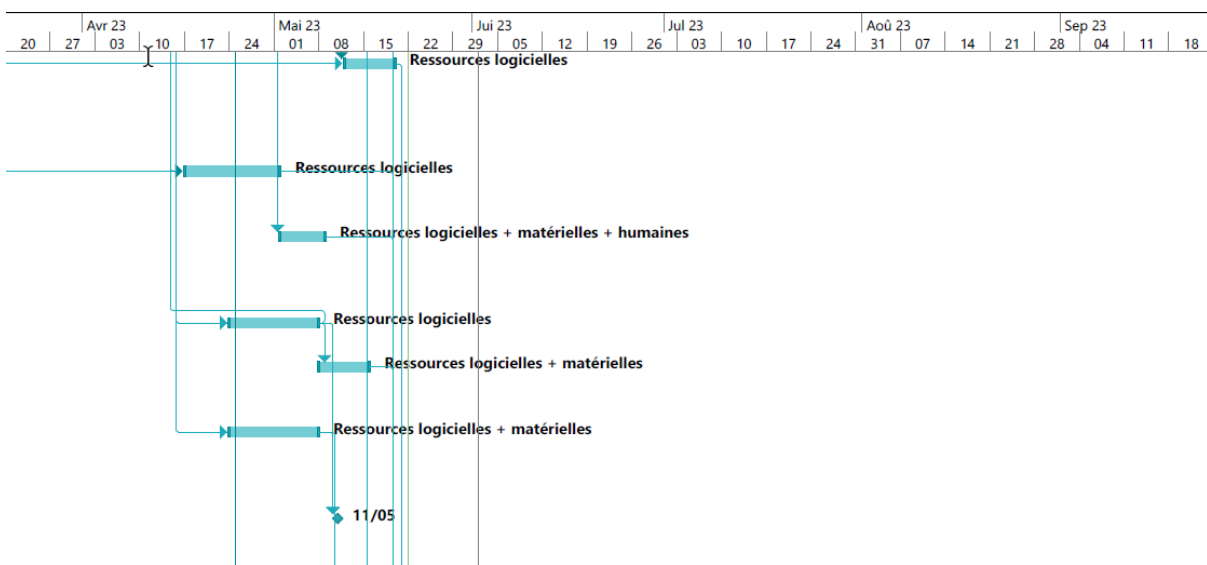
## PLANNING RÉALISÉ

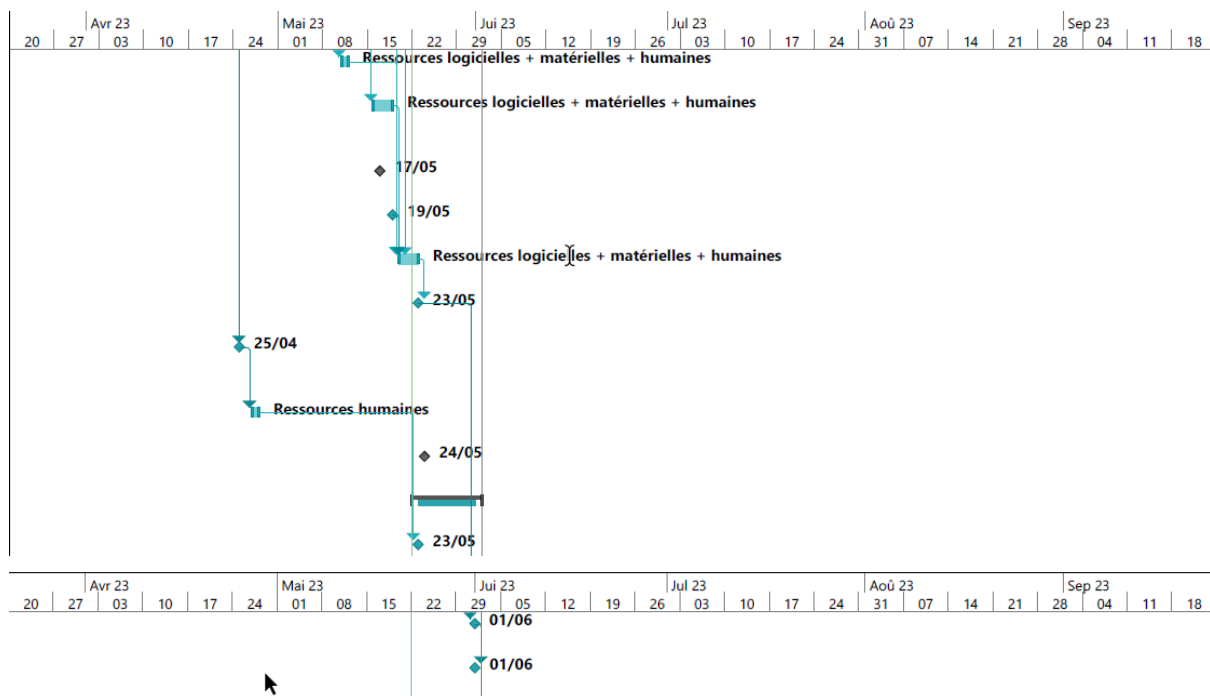
N°	Mod Tâche	Nom de la tâche	Durée	Début	Fin	Noms ressources	Responsables		Fév 23	Mar 23						
1		<b>Analyse du besoin</b>	5,6 sem	Mer 01/02/2	Ven 10/03/2			23	30	06	13	20	27	06	13	20
2		Réunion obligatoire	0 jour	Mer 01/02/23	Mer 01/02/23	Ressources humaines	Hugo Béchu		01/02							
3		Découverte du projet + recherche sur le	5 jours	Jeu 02/02/23	Mer 08/02/23	Fiche de présentation du projet + internet	Alexandre Epaillard									
4		Réunion encadrant/élèves	0 jour	Jeu 09/02/23	Jeu 09/02/23	Ressources humaines	Hugo Béchu		09/02							
5		Écriture du cahier des charges	20 jours	Ven 10/02/23	Jeu 09/03/23	Ressources humaines	Clément Jolivet									
6		Cours et TD sur l'analyse du besoin du client	1 jour	Mer 15/02/23	Mer 15/02/23	Ressources humaines	Alexandre Epaillard									
7		Réunion encadrant/élèves	0 jour	Jeu 16/02/23	Jeu 16/02/23	Ressources humaines	Hugo Béchu									
8		Formation de gestion de delais	1 jour	Mer 22/02/23	Mer 22/02/23	Logiciel MS project	Hugo Béchu									
9		Réalisation du plannig	11 jours	Jeu 23/02/23	Jeu 09/03/23	Logiciel MS project	Hugo Béchu									
10		Réunion encadrant/élèves	0 jour	Jeu 23/02/23	Jeu 23/02/23	Ressources humaines	Alexandre Epaillard									
11		Formation Git	1 jour	Mer 01/03/2	Mer 01/03/2	Logiciel Git	Clément Jolivet									
12		Formation SolidWorks	1 jour	Mer 01/03/23	Mer 01/03/23	Logiciel Solidworks	Alexandre Epaillard									

N°	Mod	Nom de la tâche	Durée	Début	Fin	Noms ressources	Responsables		Fév 23	Mar 23							
		Tâche							30	06	13	20	27	06	13	20	
13		✈ Réunion obligatoire	0 jour	Jeu 09/03/23	Jeu 09/03/23	Ressources humaines	Hugo Béchu										09/03
14		✈ Dépôt CdC + planning	0 jour	Ven 10/03/23	Ven 10/03/23	Ressources humaines	Clément Jolivet										10/03
15		✈ Conception/réalisation	61 jours	Ven 10/03/23	Jeu 01/06/23												
16		✈ Réflexions techniques	61 jours	Ven 10/03/23	Jeu 01/06/23												
17		📄 Recherche des technologies à employer	6 jours	Ven 10/03/23	Ven 17/03/23	Internet + Ressources humaines	Clément Jolivet										
18	👥	✈ Vacances	7 jours	Sam 25/03/23	Dim 02/04/23	Ressources humaines	Hugo Béchu, Alexandre Epaillard										
19	👥	✈ Formation aux technologies trouvées	49 jours	Lun 20/03/23	Jeu 01/06/23	Ressources logicielles + matérielles	Alexandre Epaillard										
20		✈ Conception	33 jours	Ven 10/03/23	Lun 24/04/23												
21		✈ Réunion obligatoire encadrant/élève	0 jour	Mer 22/03/23	Mer 22/03/23	Ressources humaines	Hugo Béchu										
22		✈ Choix du matériel	1 jour	Ven 10/03/23	Ven 10/03/23	Ressources matérielles	Clément Jolivet										
23		✈ Identifier les différentes interactions entre les différents	2 jours	Jeu 23/03/23	Ven 24/03/23	Ressources matérielles	Alexandre Epaillard										

N°	Mod	Nom de la tâche	Durée	Début	Fin	Noms ressources	Responsables	Fév 23					Mar 23			
	Tâche							23	30	06	13	20	27	06	13	2
24			Réalisation du modèle de l'application	22 jours	Sam 25/03/23	Dim 23/04/23	Ressources humaines	Hugo Béchu								
25			Réunion obligatoire encadrant/élève	0 jour	Lun 17/04/23	Lun 17/04/23	Ressources humaines	Hugo Béchu								
26			Fin de la phase de conception	0 jour	Lun 24/04/23	Lun 24/04/23	Ressources humaines	Alexandre Epaillard								
27			<b>Réalisation et tests</b>	<b>27 jours</b>	<b>Lun 17/04/23</b>	<b>Mar 23/05/24</b>										
28			Configuration de l'IHM pour permettre la déclaration de la configuration	16 jours	Lun 17/04/23	Lun 08/05/23	Ressources logicielles	Hugo Béchu								
29			Réalisation du code permettant de traiter les	2 jours	Lun 17/04/23	Mar 18/04/23	Ressources logicielles	Clément Jolivet								
30			Réalisation du code permettant d'appliquer un mode de fonctionnement	15 jours	Lun 24/04/23	Ven 12/05/23	Ressources logicielles	Hugo Béchu								

N°	Mod Tâche	Nom de la tâche	Durée	Début	Fin	Noms ressources	Responsables	23	Fév 23	Mar 23
31		Réalisation du code permettant de spécifier le mode de	6 jours	Ven 12/05/23	Ven 19/05/23	Ressources logicielles	Clément Jolivet	23	30 06 13 20	27 06 13 20
32		Réalisation du code pour piloter	11 jours	Lun 17/04/23	Lun 01/05/23	Ressources logicielles	Alexandre Epaillard			
33		Test de la bonne récupération des données	5 jours	Mar 02/05/23	Lun 08/05/23	Ressources logicielles + matérielles + humaines	Clément Jolivet			
34		Développement de l'IHM	11 jours	Lun 24/04/23	Dim 07/05/23	Ressources logicielles	Hugo Béchu			
35		Test de fonctionnement de l'IHM	6 jours	Lun 08/05/23	Lun 15/05/23	Ressources logicielles + matérielles	Hugo Béchu			
36		Elablissement des connexions entre les	11 jours	Lun 24/04/23	Dim 07/05/23	Ressources logicielles + matérielles	Alexandre Epaillard			
37		Réunion obligatoire encadrant/élèv	0 jour	Jeu 11/05/23	Jeu 11/05/23	Ressources humaines	Hugo Béchu			
N°	Mod Tâche	Nom de la tâche	Durée	Début	Fin	Noms ressources	Responsables	23	Fév 23	Mar 23
38		Test des connexions	1 jour	Jeu 11/05/23	Jeu 11/05/23	Ressources logicielles +		23	30 06 13 20	27 06 13 20
39		Test de l'IHM pour les différentes	3 jours	Mar 16/05/23	Jeu 18/05/23	Ressources logicielles + matérielles +	Clément Jolivet			
40		Réunion encadrant/élèv	0 jour	Mer 17/05/23	Mer 17/05/23					
41		Fin de la phase de réalisation	0 jour	Ven 19/05/23	Ven 19/05/23	Ressources humaines	Clément Jolivet			
42		Test global de l'application	2 jours	Sam 20/05/23	Lun 22/05/23	Ressources logicielles +	Alexandre Epaillard			
43		Fin de la phase de test	0 jour	Mar 23/05/23	Mar 23/05/23	Ressources humaines	Alexandre Epaillard			
44		QCM Moodle (atelier aide rédaction	0 jour	Mar 25/04/23	Mar 25/04/23	Ressources humaines	Clément Jolivet			
45		Atelier de rédaction du	1 jour	Jeu 27/04/23	Jeu 27/04/23	Ressources humaines	Hugo Béchu			
46		Réunion encadrant/élèves	0 jour	Mer 24/05/23	Mer 24/05/23					
47		<b>Communication écrite/restitution</b>	<b>9 jours?</b>	<b>Lun 22/05/23</b>	<b>Jeu 01/06/23</b>					
48		Dépôt du rapport	0 jour	Mar 23/05/23	Mar 23/05/23	Ressources humaine:	Alexandre Epaillard			
N°	Mod Tâche	Nom de la tâche	Durée	Début	Fin	Noms ressources	Responsables	23	Fév 23	Mar 23
49		Réunion obligatoire	0 jour?	Jeu 01/06/23	Jeu 01/06/23	Ressources humaines	Hugo Béchu			
50		Restitution finale (code, proto...)	0 jour	Jeu 01/06/23	Jeu 01/06/23	Ressources humaines	Clément Jolivet			





## ANALYSE COMPARATIVE

Rédacteur : Clément Jolivet Relecteur : Alexandre Epaillard

Le planning prévisionnel a été établi le 10 mars 2023, toutes les tâches effectuées en amont de cette date correspondent au planning réel.

Dans la tâche “Conception-réflexion technique”, la formation aux technologies trouvées s’est poursuivie tout au long du projet car nous avons eu besoin d’un apprentissage continu et nous avons découvert de nouvelles méthodes pour développer le projet. C’est notamment le cas pour la programmation qui s’est faite sous Node-RED dont nous n’avions aucune base.

L’ensemble des réunions prévues avec l’encadrant se sont tenues selon le planning prévisionnel. Nous avons tenu depuis mi-avril des réunions hebdomadaires, réunions obligatoires incluses, afin de bien avancer sur le projet.

La fin de conception du système était initialement prévue pour le 17 avril, cependant la conception par flux nécessitait plus de réflexion que la modélisation objet auxquels nous étions déjà habituée. De plus, une bonne compréhension de ce système facilite ensuite la prise en main de l’outil de programmation choisi et de la phase d’intégration.

La réalisation des codes s’est allongée en moyenne de deux semaines car il était finalement plus compliqué de programmer les fonctions exactes et précises s’alignant aux fonctionnalités

voulues. Nous n'avons cependant pas pu réaliser les codes permettant de déclarer le nombre de relais voulus, d'implémenter en fonction de Linky ainsi que son test.

La tâche 'Etablissement des connexions entre les différents composants' à été en partie réalisée. Nous avons seulement codé le programme et celui-ci a été validé avec la carte relais utilisée pour le projet en vérifiant que les bons relais étaient alimentés en fonction des données envoyées. La confirmation d'alimentation s'est faite à l'aide de voyants lumineux correspondant à chaque relais. Cependant, l'application en cas réel d'utilisation n'a pas pu être effectuée car nous n'avons pas réalisé les connexions filaires entre les composants.

L'intervention dans les plannings entre la phase de développement de l'IHM et de la récupération des données avec celle du code de fonctionnement s'explique par le fait que la prise en main de Node-RED nous semblait plus intuitive et plus applicable à notre projet si l'on débutait par l'IHM. Nous avons pu alors avoir une compréhension plus rapide de l'outil de programmation pour notre projet.

Le temps de développement du module de récupération des données de l'API d'EcoWatt s'est quant à lui vu réduit à seulement deux jours car il existait déjà de nombreux modules programmés pour cet objectif.