

Filière

Systèmes industriels

Orientation Infotronics

Rapport intermédiaire

Diplôme 2025

Marcelin Puipe

SoftPLC pour l'IoT

Professeur

HEI-Vs, Prof. Métrailler Christopher, christopher.metrailler@hevs.ch

Expert

WAGO Contact SA, Stéphane Rey, stephane.rey@wago.com

Date de soumission

17 February 2025



Informations sur ce rapport

Coordonnées

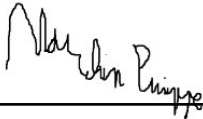
Auteur: Marcelin Puipe
Bachelor Étudiante
HEI-Vs
E-Mail: marcelin.puipe@hevs.ch

Déclaration sur l'honneur

Je soussigné(e), déclare par la présente que le travail soumis est le résultat d'un travail personnel. Je certifie ne pas avoir eu recours au plagiat ou à d'autres formes de fraude. Toutes les sources d'information utilisées et les citations d'auteurs ont été clairement mentionnées.

Lieu, date: Sion, 01.05.2025

Signature:



Contenu

Remerciements	1
Donnée de travail de bachelor	2
Développement durable	4
1 Environnement de développement	5
2 Introduction	6
3 Point de départ	6
4 Objectif	6
5 L'implémentation bloc de haut niveau	7
6 WDX	8
6.1 Accéder IO	8
6.1.1 Sans wda	8
6.1.2 Avec wda	8
7 Erreur non gérée	10
7.1 Manque lien	10
7.2 Plusieurs Output pour un Input	11
8 Ressource utile pour la suite	12
8.1 n8n [1]	12
8.2 total js [2]	13
9. Conclusion	15
9.1. Résumé du projet	15
9.1.1. Fonctionnalités développées :	15
9.1.2. changement de l'interface	15
9.2. Comparaison avec les objectifs initial	16
9.3. Difficultés rencontrées	16
9.4. Perspectives d'avenir	17
9.4.1. Idées d'amélioration et extensions du frontend web	17
10 Annexe	18
10.1 Planning	18
Bibliographie	23

Remerciements

Je tiens à remercier mon répondant de travail de bachelor, M. Christopher Métrailler ainsi que son assitant Michael Clausen qui ont répondu à mes différentes questions, pour leurs conseils et pour avoir pris de leur temps pour mon travail. Je remercie aussi Arnaud Ducrey pour son TB documenté.

Donnée de travail de bachelor

HES-SO Valais

SYND	ETE	TEVI
X	X	X



Données du travail de bachelor Aufgabenstellung der Bachelorarbeit

FO 1.2.02.07.FB
che/13.03.2024

Filière / Studiengang SYND	Année académique / Studienjahr 2024-25	No TB / Nr. BA IT/2025/5
Mandant / Auftraggeber <input type="checkbox"/> HES—SO Valais-Wallis <input checked="" type="checkbox"/> Industrie <input type="checkbox"/> Etablissement partenaire Partnerinstitution	Etudiant-e / Student/in Marcelin Puippe Professeur-e / Dozent/in Christopher Métrailler	Lieu d'exécution / Ausführungsort <input checked="" type="checkbox"/> HES—SO Valais-Wallis <input type="checkbox"/> Industrie <input type="checkbox"/> Etablissement partenaire Partnerinstitution
Travail confidentiel / vertrauliche Arbeit <input type="checkbox"/> oui / ja <input checked="" type="checkbox"/> non / nein	Expert-e / Experte/Expertin (nom, prénom, E-Mail / Name, Vorname, E-Mail) Stéphane Rey, WAGO Contact SA, stephane.rey@wago.com	

Titre SoftPLC pour IoT
<p>Description</p> <p>L'entreprise WAGO commercialise des automates programmables industriels basés Linux, également équipé d'un environnement SoftPLC basé Codesys. Un PoC d'un environnement SoftPLC basé Web a été développé dans un précédent projet. Ce travail de diplôme consiste à développer un nouveau HAL permettant d'intégrer les nouvelles interfaces des PLC WAGO CC100 (751-9401 et 751-9402), tel qu'une interface CAN. L'environnement SoftPLC doit être complété afin d'intégrer des modules I/O plus complexes. De nouveaux composants doivent être développés afin d'utiliser ces derniers de manière graphique dans le SoftPLC. Ces développements doivent permettre d'intégrer aisément de nouvelles interfaces et capteurs au PLC WAGO dans le futur, comme des IoT disponibles dans des maisons connectées (CAN, HTTP/MQTT, WiFi, etc.). Un banc de test équipé de plusieurs capteurs permettra de valider le bon fonctionnement des développements.</p> <p>Objectifs du projet individuel (Pr4)</p> <ul style="list-style-type: none"> — Étude et analyse du projet SoftPLC existant. Prise en main de l'environnement de développement et des outils Docker pour le déploiement d'application sur les PLC Wago CC100 — Développement d'applications de démonstration basées Golang (backend) et Javascript/TypeScript (frontend) pour le système existant en utilisant le banc de test existant comme base <p>Objectifs du travail de Bachelor (TB)</p> <ul style="list-style-type: none"> — Modification du Hardware Abstraction Layer (HAL) pour les nouveaux automates CC100. Développement d'un nouveau HAL pour les I/O et le module CAN en utilisant l'API VDX — Définition et implémentation de nouveaux blocs SoftPLC comprenant des blocs de haut niveau, tels que client MQTT, CAN, WebServer, client/serveur HTTP, etc. Extension du système SoftPLC existant et de l'interface web pour permettre l'ajout de ces nouveaux composants — Amélioration et extensions du frontend web avec contrôle du type d'entrées/sorties, des types des signaux. Amélioration de l'interface et de l'expérience utilisateur avec un meilleur visuel et des contrôles automatisés — Développement d'un banc de test physique et d'une application de démonstration pour une maison connectée. Documentation et tests et rédaction du rapport, poster et présentation.

Délais / Termine	
Démarrage du projet individuel (Pr4) <i>Start des individuellen Projekts (Pr4)</i> 17.02.2025 Présentation du projet individuel (Pr4) <i>Präsentation:des individuellen Projekts (Pr4)</i> 02.05.2025 Démarrage du travail de bachelor <i>Start der Bachelorarbeit:</i> 19.05.2025	Remise du rapport final / <i>Abgabe des Schlussberichts:</i> 14.08.2025, 12:00 Expositions et Pitch / <i>Ausstellungen und Pitch der Bachelorarbeiten:</i> 22.08.2025 – HEI 25.08.2025 – Monthey 28.08.2025 – Visp Défense orale / <i>Mündliche Verfechtung:</i> Semaines/Wochen 36-37 (01-12.09.2025)

Signature ou visa / Unterschrift oder Visum	
Responsable de l'orientation / Leiter/in der Vertiefungsrichtung: 	¹ Etudiant·e / Student/in: 

¹ Par sa signature, l'étudiant·e s'engage à respecter strictement la directive DI.1.2.02.07 « Travail de bachelor ».
 Durch seine Unterschrift verpflichtet sich er/die Student/in, sich an die Richtlinie DI.1.2.02.07 „Bachelorarbeit“ zu halten.

Développement durable

Traduction de l'anglais par chatgpt [3] :

L'Agenda 2030 pour le développement durable, adopté par l'ensemble des États membres des Nations Unies en 2015, constitue une feuille de route commune pour la paix et la prospérité des peuples et de la planète, aujourd'hui et pour l'avenir. Au cœur de cet agenda se trouvent les 17 Objectifs de Développement Durable (ODD) [4], un appel urgent à l'action lancé à tous les pays – développés comme en développement – dans le cadre d'un partenariat mondial. Ces objectifs reconnaissent que l'éradication de la pauvreté et d'autres privations doit aller de pair avec des stratégies visant à améliorer la santé et l'éducation, à réduire les inégalités et à stimuler la croissance économique – tout en luttant contre les changements climatiques et en œuvrant à la préservation des océans et des forêts.



Fig. 1. – 17 Objectifs de Développement Durable (ODD) [4]

Le projet consiste en la création d'un HAL de développement d'automate. Il est donc très probable que le projet soit utiliser dans des applications allant dans le sans d'un développement durable. En effet, l'automatisation des processus industriels peut contribuer à la durabilité en améliorant l'efficacité énergétique (ODD 7), en réduisant les déchets et en optimisant l'utilisation des ressources. De plus, l'automatisation peut également aider à surveiller et à contrôler les émissions de gaz à effet de serre, ce qui contribue à la lutte contre le changement climatique. L'autiamisation favorise également la croissance économique (ODD 8) en augmentant la productivité et en réduisant les coûts de production. En outre, l'automatisation peut également améliorer la sécurité au travail (ODD 3) en réduisant le risque d'accidents liés à des tâches dangereuses.

Cependant, le programme peut également être utilisé pour des applications moins durables. Par exemple, il peut être utilisé pour automatiser des processus industriels qui ont un impact environnemental lourd (ODD 13), comme l'extraction de combustibles fossiles ou la production de déchets toxiques (ODD 14-15).

1 | Environnement de développement

Appareil	Adresse IP	Utiliser dans soft
Automate WAGO CC100	192.168.37.134	softplc-main
PC	localhost	softplcui-main

La salle 23.N320 utilisé pour connecté l'automate WAGO CC100 sur le réseau.

Logiciel	Appareil	Commentaires
Goland	PC	Développement logiciel
Umlet	PC	Réalisation de schéma basé développement
Firefox	PC	Permettant meilleure visualisation de WDX
WDX	Automate	Dénomination générale pour parler de WDM + WDD + WDA : <ul style="list-style-type: none"> • WDM = WAGO Device Model (standard utilisé pour les WDD) • WDD = WAGO Device Description (manifeste décrivant ce que le produit met à disposition) • WDA = WAGO Device Access (accès aux paramètres et IO)
WDA	Automate	Nouvelle librairie, interface REST, WAGO accessible par web en JSON. Permet la récupération des informations de l'automate et le contrôle des entrées/sorties par requête JSON. https://192.168.1.126/wda/parameters?page[limit]=20000 https://192.168.1.126/wda/parameter-definitions?page[limit]=20000

2 | Introduction

Ce document présente les travaux réalisés ainsi que les perspectives à venir pour le projet PLCSoft destiné à WAGO. Ce projet consiste en l'ajout de nouvelles fonctionnalités à un HAL (Hardware Abstraction Layer) pour le développement d'automates via une interface web. La programmation se fait de manière graphique et modulaire sur une page web, en reliant les différentes fonctions disponibles entre elles. L'objectif principal du projet est d'ajouter de nouveaux blocs fonctionnels pouvant être utilisés dans ce cadre de développement.

3 | Point de départ

Le projet se construit sur la base de deux programmes déjà développés, lors du TB 2024 :

- **Softplcui-main** : Gérant l'interface web côté PC.
- **Softplc-main** : Gérant l'activation des entrées / sorties selon le programme build depuis PLC UI.

Le travail effectué précédemment nous prouve la faisabilité du développement d'un tel HAL. Les fonctionnalités implémentées dans par ce précédent projet sont :

- Digital Input / output
- Analogue Input / output
- Ton (timer retardé à la montée)

4 | Objectif

L'objectif est l'amélioration et l'implémentation de nouvelles fonctionnalité. Les tâches devant être réalisées sont :

- Modifier les programmes actuelle pour utiliser la nouvelle interface REST WDA pour piloté les nouveaux automates.
- L'implémentation de nouveaux blocs de haut niveau comme CAN, MQTT, Web-Server, client/serveur HTTP et autres bloc. Il faudra trouver une solution pour faire ces tâches par programmation en bloc.
- Amélioration et extensions du frontend web.
- Développement d'un banc de test physique et d'une application de démonstration pour une maison connectée.
- Documentation et tests et rédaction du rapport, poster et présentation.

5 | L'implémentation bloc de haut niveau

Il existe plusieurs manières d'aborder le problème. Une des approches est de repérer les points communs entre ces blocs de haut niveau pour essayer d'en tirer une forme générique. On remarque que tous ces blocs ont pour objectif de transmettre et recevoir des données. Il faudra donc commencer par le développement de bloc commun pour une communication. Il faut également des blocs permettant de travailler avec des STRING. Le schéma figure 1 montre le concept d'une telle structure avec tous les blocs qui devront être développés autour pour pouvoir créer une communication.

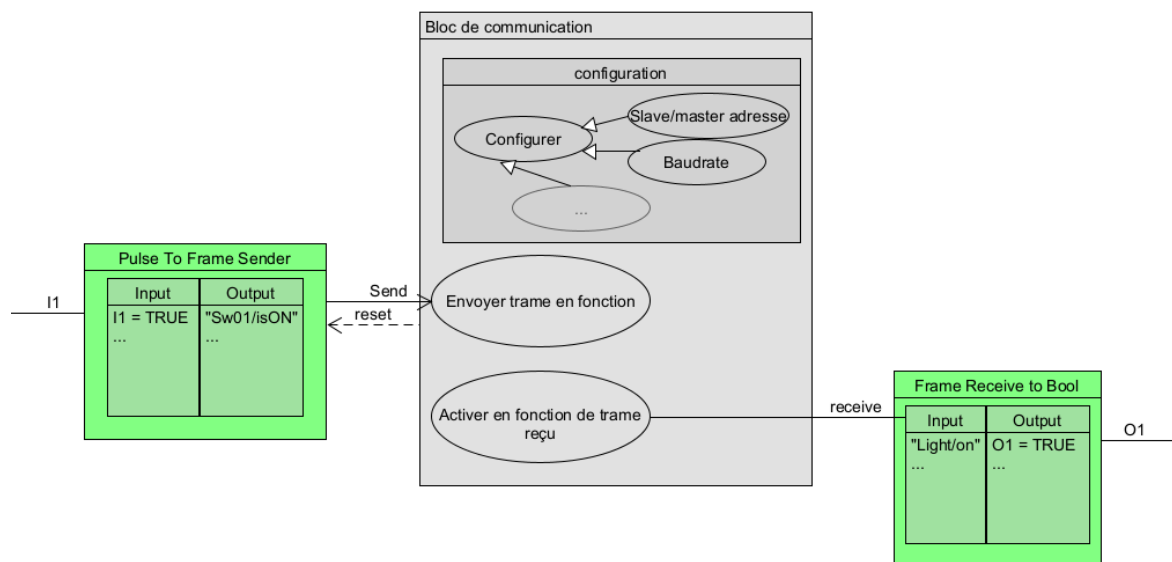


Fig. 2. – Communication principe



L'idée étant d'avoir un bloc communication qui s'occupe de la configuration étant différente pour can, Mqtt etc. Sur le quel, on pourra double cliqués pour accéder à la page de configuration. Sur ce bloc de communication, on pourrait ensuite venir lier nos 2 blocs permettant la transition de boolean vers nos trame. Par le future, en mode debug, l'utilisateur pourra voir l'état de la communication grâce au « bloc de communication » et voir ce que la logique combinatoire transmet comme trame grâce au bloc en vert.

6 | WDX

Documentation API : <https://192.168.37.134/openapi/wda.openapi.html>

Documentation : <https://downloadcenter.wago.com/wago/null/details/m2ddbfiuhrn44rp2h>

Json Parameter : [https://192.168.37.134/wda/parameter-definitions?page\[limit\]=20000](https://192.168.37.134/wda/parameter-definitions?page[limit]=20000)

6.1 Accéder IO

6.1.1 Sans wda

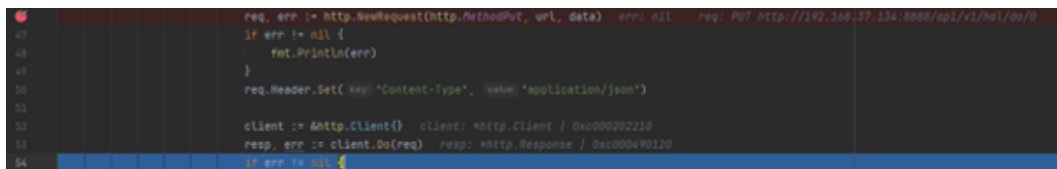
Lien pour accéder sur page web : 192.168.37.134:8888/api/v1/hal/io

Résultat affiché sur la page web :

```
{« di »:[false,false,false,false,false,false,false,false],« do »:[false,false,false,false],« ai »:[0.336,0.343],« ao »:[0,0],« temp »:[16778.26508951407,16778.26508951407]}
```

Activer output :

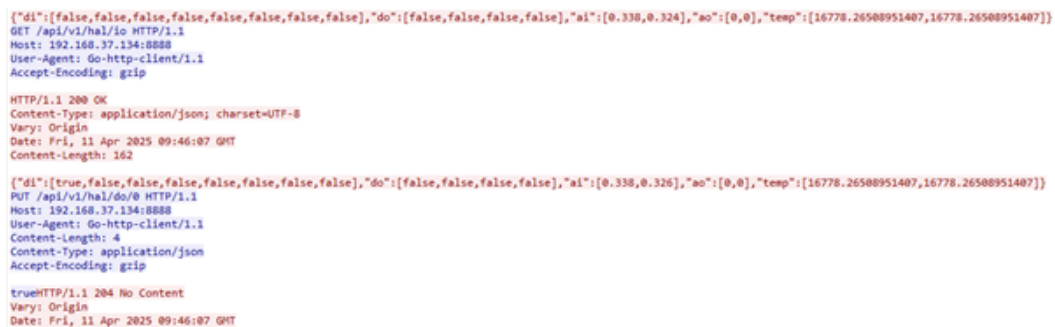
Dans le fichier « OutputUpdate.go » de softplc-main. Pour active DO1 : PUT <http://192.168.37.134:8888/api/v1/hal/do/0>



```
47 req, err := http.NewRequest(http.MethodPut, url, data)  err: nil  req: PUT http://192.168.37.134:8888/api/v1/hal/do/0
48 if err != nil {
49     fmt.Println(err)
50 }
51 req.Header.Set("Content-Type", "application/json")
52
53 client := &http.Client{}  client: *http.Client / 0xc000202210
54 resp, err := client.Do(req)  resp: *http.Response / 0xc000490120
55 if err != nil {
```

Fig. 3. – programme : OutputUpdate.go

C'est à partir de la ligne 54 qu'on a la lampe allumée. Plus de détaille dans dossier « autre » puis dossier « Request ».



```
[{"di":true,"do":false,"ai":0.336,"ao":0,"temp":16778.26508951407}]
GET /api/v1/hal/io HTTP/1.1
Host: 192.168.37.134:8888
User-Agent: Go-http-client/1.1
Accept-Encoding: gzip

HTTP/1.1 200 OK
Content-Type: application/json; charset=UTF-8
Vary: Origin
Date: Fri, 11 Apr 2025 09:46:07 GMT
Content-Length: 162

{"di":true,"do":false,"ai":0.336,"ao":0,"temp":16778.26508951407}]
PUT /api/v1/hal/do/0 HTTP/1.1
Host: 192.168.37.134:8888
User-Agent: Go-http-client/1.1
Content-Length: 4
Accept-Encoding: gzip

trueHTTP/1.1 204 No Content
Vary: Origin
Date: Fri, 11 Apr 2025 09:46:07 GMT
```

Fig. 4. – wireShark Http stream

6.1.2 Avec wda

751-9402 : <https://192.168.37.134/wda/parameters/0-0-io-channelcompositions-1-channels>

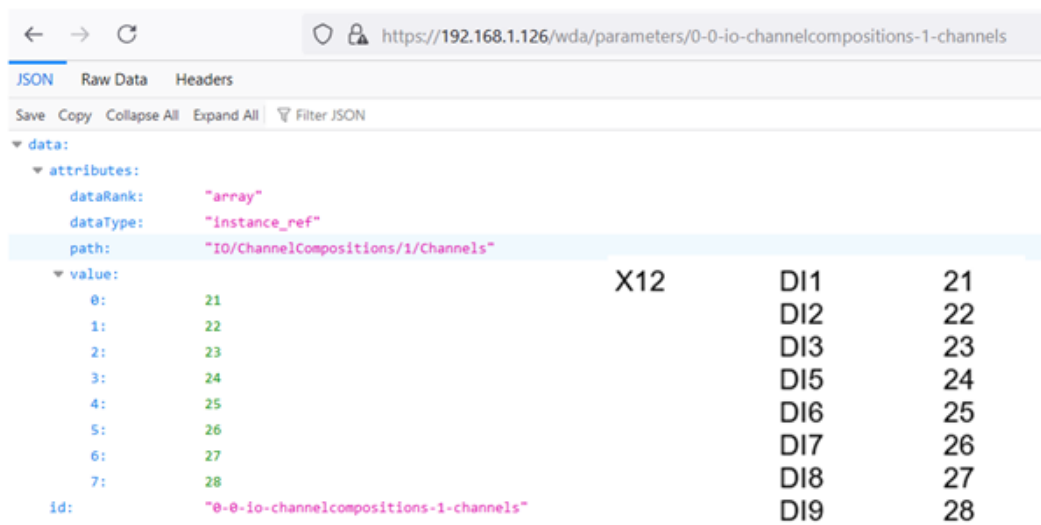


Fig. 5. – automate 751-9402 accéder aux IO

C'est possible avec un automate 751-9402 [5], cependant on a le modèle 751-9401 [6]. Cela ne signifie pas que c'est impossible, mais que ce n'est pas documenté.

7 | Erreur non gérée



Certaines erreurs n'ont pas été traitées lors de l'ancien TB. Cette section présente ces erreurs qui devront être réglées.

7.1 Manque lien

Le problème est que le programme plcSoft plante au lieu d'afficher simplement une erreur et de ne pas Build le programme dans l'automate.

Cependant, le save est possible et le restore peut être fait après avoir relancé le programme.

Remarque Bloc bleu : il y a des blocs bleus qui sont le résultat d'un test d'explication dans la synthèse.

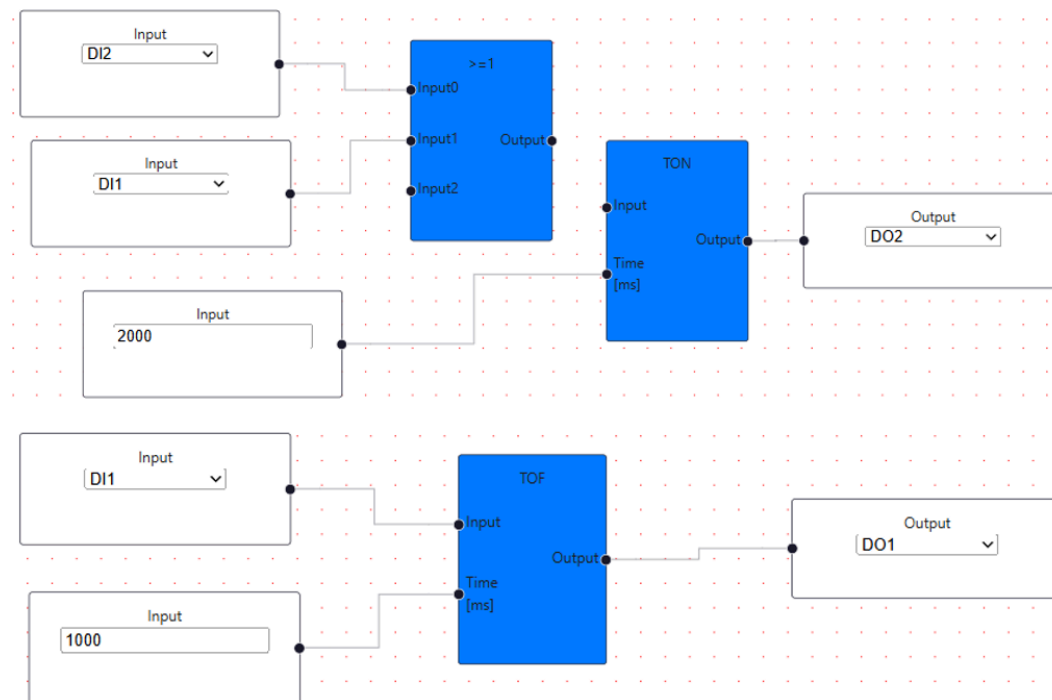


Fig. 6. – Défaut manque lien entre OR et TON

```

OutputNodes:
&{3 digitalOutput [{ D01 {0xc00026f050 Input bool}}]}
&{6 digitalOutput [{ D02 {0xc00026f140 Input bool}}]}
LogicalNode:
&{0 TOFNode [{0xc000228018 Input bool} {0xc00046ae78 Time [ms] value}] [{0 Output bool}] {<nil> false} false 0 false}
&{7 TONNode [{<nil> Input bool} {0xc00041cbb8 Time [ms] value}] [{0 Output bool}] {<nil> false} false 0 false}
InputNodes:
&{2 digitalInput [{ DI1 {0xc000228018 Output bool}}]}
&{4 constantInput [{ {0xc00046ae78 Output value}}]}
&{8 constantInput [{ {0xc00041cbb8 Output value}}]}
Const:
{4 1000}
{8 2000}

```

Fig. 7. – message plcsoft save défaut

```

panic: runtime error: invalid memory address or nil pointer dereference
[signal 0xc0000005 code=0x0 addr=0x0 pc=0xc1a412]

goroutine 1 [running]:
SoftPLC/nodes.(*TONNode).ProcessLogic(0xc0001813b0)
    C:/Users/puipp/Desktop/TB_PLCSoft_Wago/Software/Go/softplc-main/softplc-main/nodes/tonNode.go:89 +0x1d2
main.main()
    C:/Users/puipp/Desktop/TB_PLCSoft_Wago/Software/Go/softplc-main/softplc-main/softPLC.go:31 +0x13c

Process finished with the exit code 2

```

Fig. 8. – Erreur Build manque lien entre OR et TON

7.2 Plusieurs Output pour un Input



Cela devrait être faisable, pourtant c'est interdit.

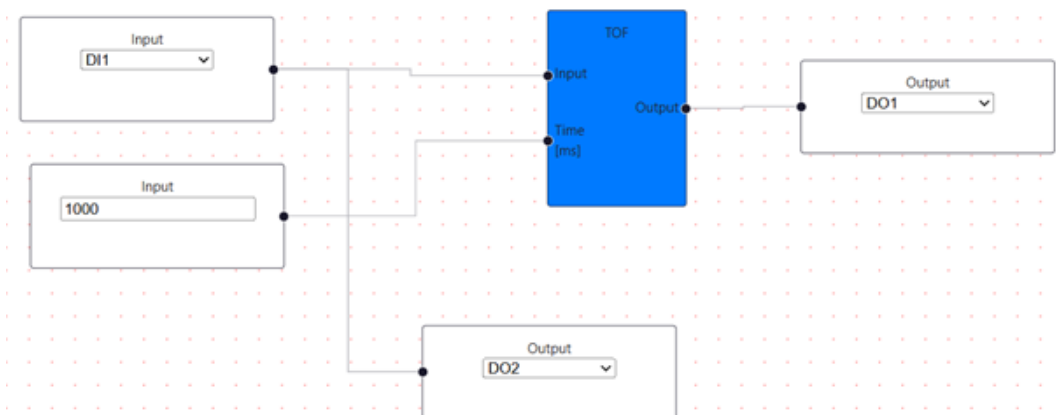


Fig. 9. – Erreur 2 output pour un input, avec une output directement sur l'input

8 | Ressource utile pour la suite

Durant le cours projet 4, plusieurs ressources pouvant être utiles pour la suite ont été trouvées.

8.1 n8n [1]

C'est un logiciel d'automatisation présent en ligne sur GitHub. Il permet une programmation en no-code sur page web comme ce qu'on l'on essaie de faire. Il est surtout conçu pour l'automatisation de tâche simple. Il permet notamment l'automatisation de chat alimenté par l'IA, c'est-à-dire des réponses automatiques. Ce n'est pas ce qui nous intéresse mais cela peut nous aider à avoir des idées.

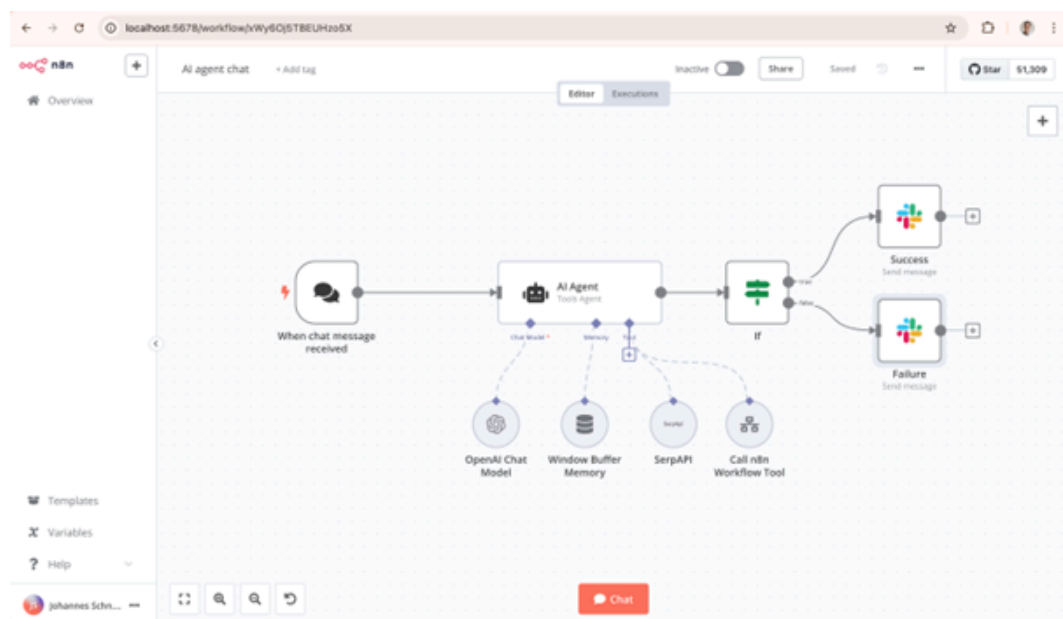


Fig. 10. – Exemple gitHub n8n

L'activation d'une output en n8n peut se faire de la manière suivante. Il suffit d'un bloc HTTP Request1 qu'on configure.

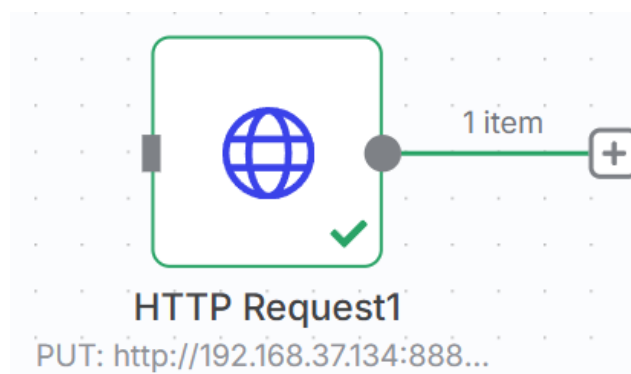


Fig. 11. – Bloc HTTP Request1 en n8n

The image shows the configuration interface for the 'HTTP Request1' node in n8n. The interface is divided into two tabs: 'Parameters' (selected) and 'Settings'. A red 'Test step' button is in the top right corner. The 'Parameters' tab contains the following fields and controls:

- Method:** A dropdown menu set to 'PUT'.
- URL:** A text input field containing 'http://192.168.37.134:8888/api/v1/hal/do/1'.
- Authentication:** A dropdown menu set to 'None'.
- Send Query Parameters:** A toggle switch that is currently turned off.
- Send Headers:** A toggle switch that is currently turned on (green).
- Specify Headers:** A dropdown menu set to 'Using Fields Below'.
- Header Parameters:** A section with two input fields: 'Name' (containing 'Content-Type') and 'Value' (containing 'application/json'). Below these is an 'Add Parameter' button.
- Send Body:** A toggle switch that is currently turned on (green).
- Body Content Type:** A dropdown menu set to 'JSON'.
- Specify Body:** A dropdown menu set to 'Using JSON'.
- JSON:** A text area containing a single JSON object: `{ "1": "true" }`.
- Options:** A section with the text 'No properties' and an 'Add option' button.

At the bottom of the configuration area, there is a yellow informational box that reads: 'You can view the raw requests this node makes in your browser's developer console'.

Fig. 12. – configuration HTTP Request1 en n8n pour activation output 2

8.2 total js [2]

c'est un logiciel de programmation en no-code. Il est possible de faire des programmes en JS, mais il y a aussi une interface graphique.

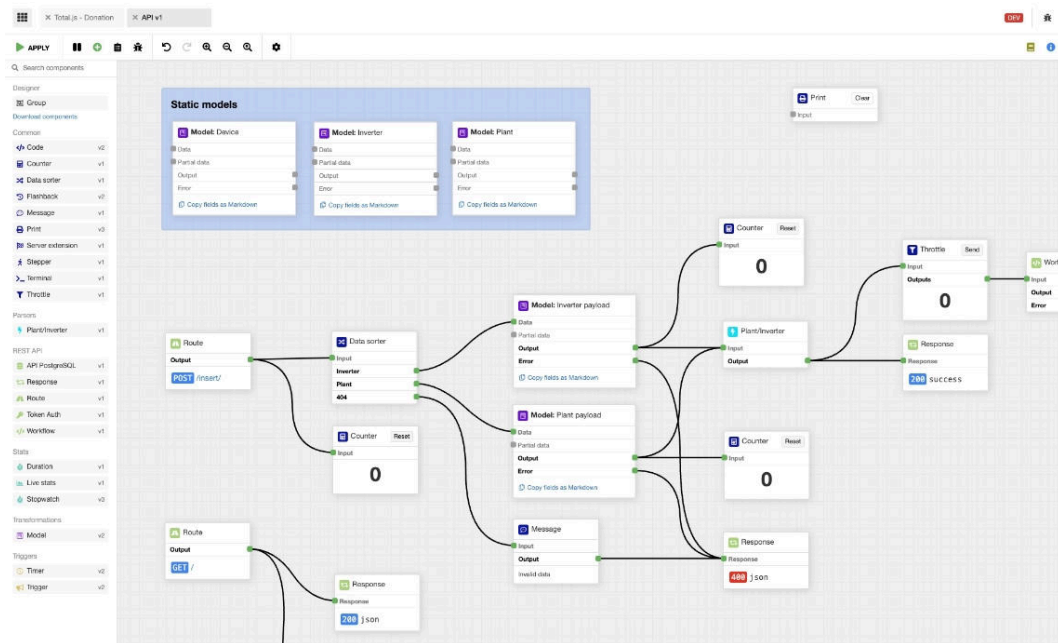


Fig. 13. – Exemple totalJS

9. Conclusion

9.1. Résumé du projet

L'automate a été câblé et configuré. Il est prêt à être utilisé pour la suite du projet. Les programmes softplc-main et softplcui-main ont pu être testés et fonctionnent comme décrit dans le travail précédent de TB. Toutefois, la partie analogique n'a pas été testée, mais elle ne semble pas fonctionnelle, car elle ne figurait pas parmi les points traités dans le TB précédent.

Par ailleurs, le bloc Appliance Input ne fonctionne pas et fait planter l'interface. De nombreuses améliorations, décrites dans la partie « Chapitre 9.4.1 », restent possibles.

9.1.1. Fonctionnalités développées :

- Un nouveau bloc timer de type **TOR** a été ajouté. Cela prouve la faisabilité de l'ajout de blocs simples et montre que le programme est à la fois robuste et adaptable.
- Ajout de la fonctionnalité de sauvegarde/restauration via un fichier Très pratique, elle évite de devoir réécrire le code à chaque modification ou chargement du programme.
- Ajout d'une **SlideBar** [7], une fonctionnalité nécessaire pour l'ergonomie de l'interface, permet de voir les éléments plus bas.

Il a été décidé de concentrer les efforts sur l'ajout de fonctionnalités permettant de gagner du temps lors du développement et des tests.

9.1.2. changement de l'interface

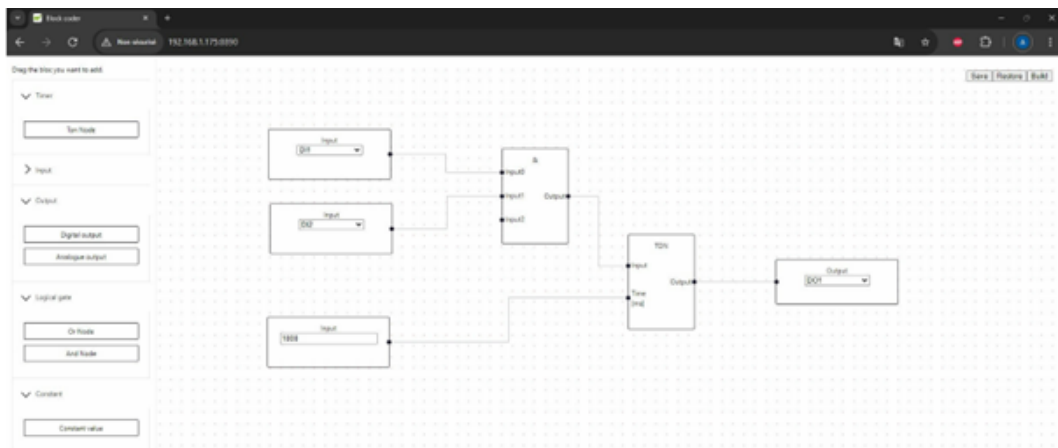


Fig. 14. – Interface avant

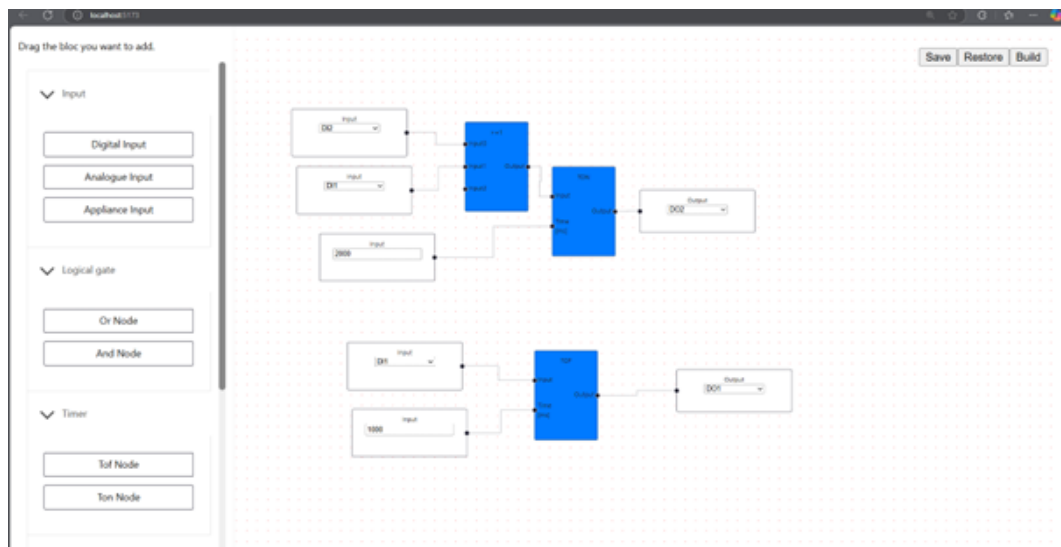


Fig. 15. – Interface après

La différence est la **slide Bar** car avant si on ouvrait tous on n'avait pas accès aux composants du bas. On voit également le nouveau bloc de type **TOR** qui a été ajouté.



Bloc bleu : il sont le résultat d'un test qui a été fait l'objectif était de voir comment était géré le style css des blocs. Le résultat est qu'il est géré par groupe. Ainsi, tous les Blocs *LogicalNode* ont le même type. Il faudra donc améliorer la structure pour rendre plus facile l'attribution de style si on veut plus personnaliser.

9.2. Comparaison avec les objectifs initial

Les objectifs fixés par pr4 sont atteints. Le programme a pu être testé et permet de créer des programmes très simples. Un nouveau bloc a été ajouté et testé sur l'automate. Le principe de fonctionnement des codes a été vu et il est possible d'ajouter de nouveaux. Cependant, le programme n'est pas parfait. Il reste des points à améliorer, notamment des erreurs .

9.3. Difficultés rencontrées

La documentation de WDA n'est pas suffisante pour comprendre le fonctionnement de la *library*. Il y a beaucoup de paramètres différents, mais on ne trouve pas ceux qui nous intéressent, la majorité d'entre eux sont pour modifier des paramètres de la configuration automate. Cependant, il a pu être remarqué que les modèles **741-9402** et **751-9401** ne sont pas les mêmes. La documentation du 741-9402 est plus complète, et l'utilisation des entrées/sorties (I/O) y est clairement expliquée. En revanche, il n'a toujours pas été trouvé de documentation concernant l'utilisation du module CAN.

9.4. Perspectives d'avenir

9.4.1. Idées d'amélioration et extensions du frontend web

Il y a de nombreuses possibilités d'amélioration pour l'interface utilisateur.

- Interdire les liens qui passent sur un bloc, ajouter une intelligence de connexion.
- Interdire les boucles de rétroaction (comme dans Codesys) ou les gérer proprement.
- Ajouter des blocs logiques contenant un champ (pour les Inputs, c'est déjà en partie fait, mais non fonctionnel, et il n'y a pas de système de seuil).
- Améliorer la nomenclature : éviter d'utiliser « Output » pour l'analogique et le digital, et « Input » pour les constantes. Une idée serait d'ajouter un menu déroulant sur le bloc pour choisir le type.
- Permettre de coder le frontend indépendamment du backend, c'est-à-dire générer les accordéons à partir d'un fichier, qui peut être mis à jour lorsqu'on est connecté.
- Ajout de raccourcis clavier :
 - Rendre la touche *Delete* fonctionnel.
 - Ctrl + C / V / A / Z / Y.
 - Clic + glisser = multi-sélection.
 - Touche O pour placer un Output, I pour un Input.
 - Touche Espace pour placer un composant identique au précédent.
 - Shift + clic gauche + glisser pour dupliquer.
 - Une autre idée intéressante : une touche (Ctrl + Alt + C) pour ajouter automatiquement tous les blocs nécessaires autour d'un bloc ou groupe sélectionné, avec des valeurs par défaut. Par exemple, on sélectionne un bloc TON, on appuie sur la touche, et le système ajoute automatiquement une constante de 1 seconde, une entrée DIO1 (ou la suivante si déjà utilisée), et une sortie DO1. Les valeurs par défaut ne sont pas obligatoires, on peut faire sans.
 - Touche dédiée pour activer ou désactiver les valeurs par défaut.
- L'ordre des Inputs, Outputs, blocs logiques, etc. dans l'accordion n'est jamais le même, ce qui rend l'utilisation plus pénible car on ne peut pas s'habituer.
- Amélioration de l'aspect visuel : couleurs et autres éléments graphiques.
- Ajout d'une barre de menu en haut :
 - Affichage des raccourcis clavier
 - Aide
 - Choix de l'emplacement d'enregistrement par l'utilisateur
 - Ajout d'un mode Debug

10 | Annexe

10.1 Planning

PLCSoft

HES-SO

Marcelin Puipe

Durée Total [j] : 62

Date de début du projet : 19.05.2025 Date de fin : 14.08.2025

Incrément de défilement : 0

Légende :

En bonne voie

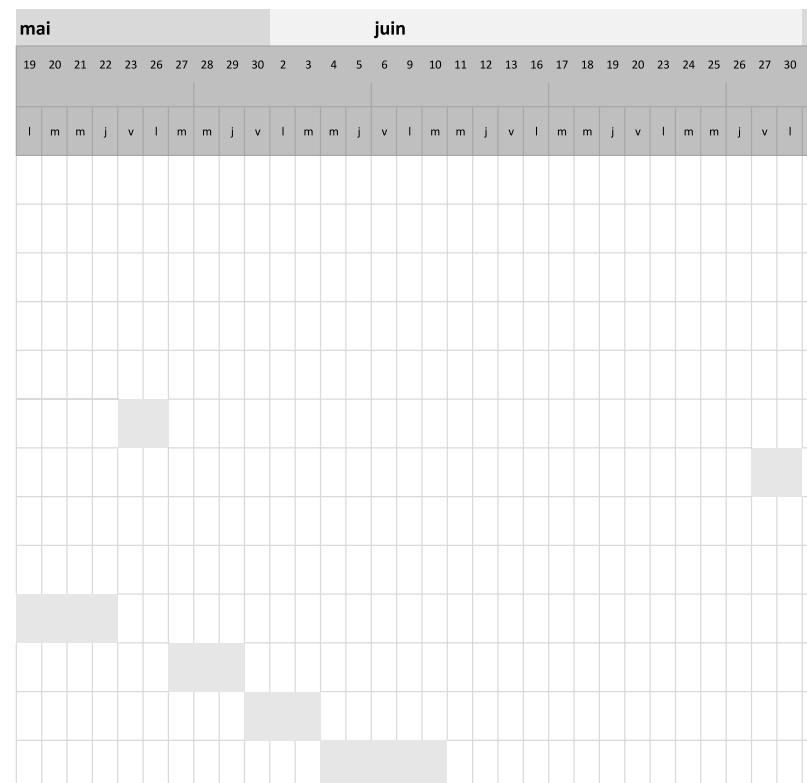
Risque faible

Risque moyen

Risque élevé

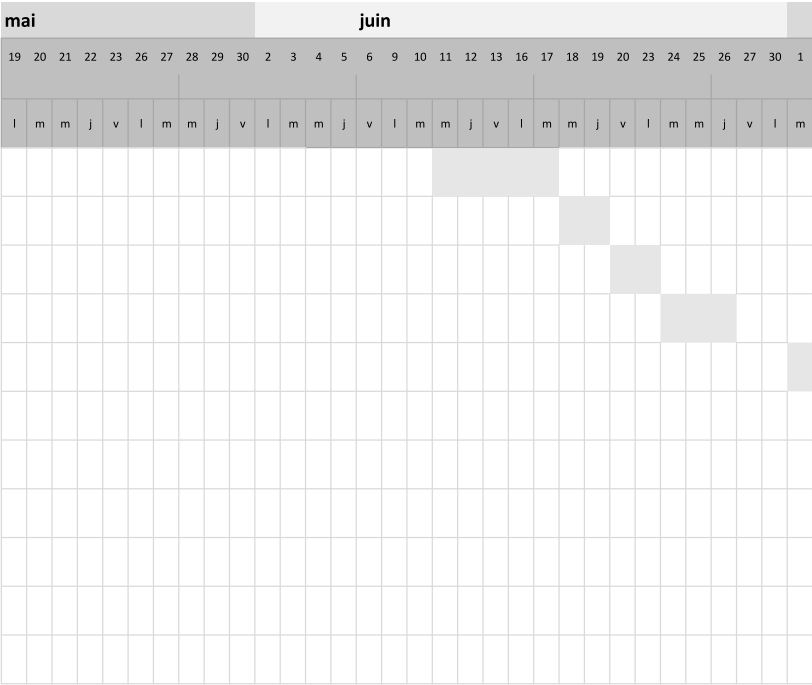
Non attribué

Description du jalon	Catégorie	Progression	Début	Jours
Dates importante				
Présentation du projet individuel	Objectif	0%	02.05.2025	1
Remise du rapport final	Objectif	0%	14.08.2025	1
Expositions et Pitch	Objectif	0%	22.08.2025	1
Documentation				
Conception architecture général		0%	23.05.2025	2
Conception détailler bloc communication		2%	27.06.2025	2
Rapport : Description des fonctionnalités, Protocole de test, Description utilisateurs		5%		
Programmation		0%		
Correction des erreurs du TB 2024		0%	19.05.2025	4
Amélioration Visu, ordre dans Accordion, affichage des Nodes, nomenclature		0%	27.05.2025	3
Ajout menu + ajout gestion de fichier		0%	30.05.2025	3
Ajout de raccourcis clavier		0%	04.06.2025	5



Date de début du projet : 19.05.2025 Date de fin : 14.08.2025
Incrément de défilement : 0

Description du jalon	Catégorie	Progression	Début	Jours
Ajout mode Debug		0%	11.06.2025	5
Création bloc "TRIG, R_TRIG, F_TRIG"		0%	18.06.2025	2
Création bloc logiques contenant un champ + système de seuil		0%	20.06.2025	2
Création bloc "bool to frame"		0%	24.06.2025	3
Création bloc "bloc de communication MQTT"		0%	01.07.2025	10
Création bloc "bloc de communication HTTP"		0%	15.07.2025	5
Création bloc "bloc de communication WebServer"		0%	22.07.2025	5
Création bloc "bloc de communication CAN"		0%	29.07.2025	8
Debugage			08.08.2025	
Tester sur maison connectée, application de démonstration			08.08.2025	2
Reserve			12.08.2025	1



PLCSoft

HES-SO

Marcelin Puippe

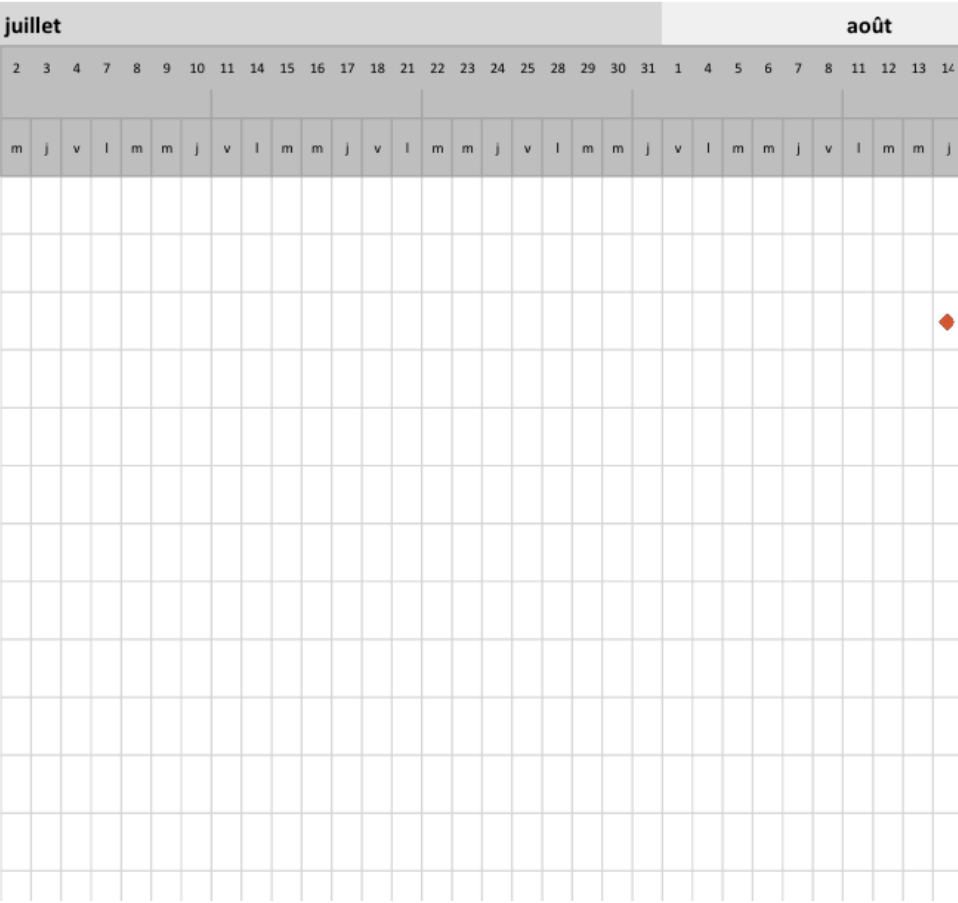
Date de début du projet : 19.05.2025
Incrément de défilement : 32

Durée Total [j] : 62
Date de fin : 14.08.2025

Légende :

- En bonne voie
- Risque faible
- Risque moyen
- Risque élevé
- Non attribué

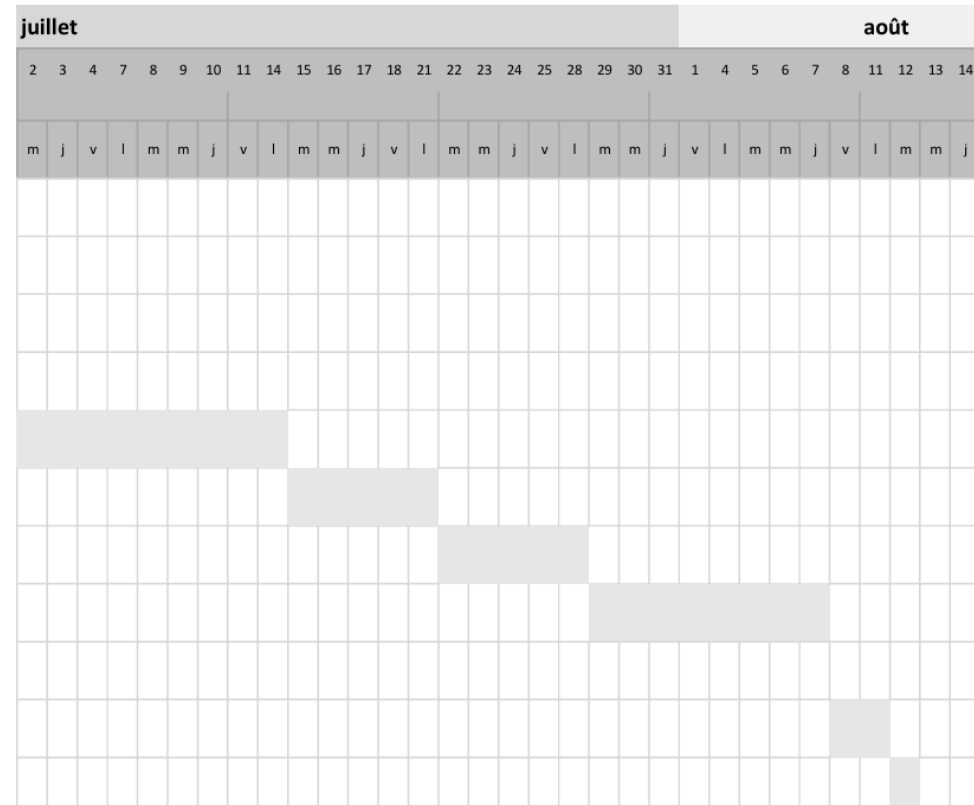
Description du jalon	Catégorie	Progression	Début	Jours
Dates importante				
Présentation du projet individuel	Objectif	0%	02.05.2025	1
Remise du rapport final	Objectif	0%	14.08.2025	1
Expositions et Pitch	Objectif	0%	22.08.2025	1
Documentation				
Conception architecture général		0%	23.05.2025	2
Conception détailler bloc communication		2%	27.06.2025	2
Rapport : Description des fonctionnalités, Protocole de test, Description utilisateurs		5%		
Programmation				
Correction des erreurs du TB 2024		0%	19.05.2025	4
Amélioration Visu, ordre dans Accordion, affichage des Nodes, nomenclature		0%	27.05.2025	3
Ajout menu + ajout gestion de fichier		0%	30.05.2025	3
Ajout de requêtes studies		0%	04.06.2025	5



Date de début du projet : 19.05.2025 Date de fin : 14.08.2025

Incrément de défilement : 32

Description du jalon	Catégorie	Progression	Début	Jours
Ajout mode Debug		0%	11.06.2025	5
Création bloc "TRIG, R_TRIG, F_TRIG"		0%	18.06.2025	2
Création bloc logiques contenant un champ + système de seuil		0%	20.06.2025	2
Création bloc "bool to frame"		0%	24.06.2025	3
Création bloc "bloc de communication MQTT"		0%	01.07.2025	10
Création bloc "bloc de communication HTTP"		0%	15.07.2025	5
Création bloc "bloc de communication WebServer"		0%	22.07.2025	5
Création bloc "bloc de communication CAN"		0%	29.07.2025	8
Debugage			08.08.2025	
Tester sur maison connectée, application de démonstration			08.08.2025	2
Reserve			12.08.2025	1



Bibliographie

- [1] « N8n-Io/N8n ». Consulté le: 27 avril 2025. [En ligne]. Disponible sur: <https://github.com/n8n-io/n8n>
- [2] « JavaScript Libraries and Components for Web Development ». Consulté le: 27 avril 2025. [En ligne]. Disponible sur: <https://www.totaljs.com/>
- [3] « ChatGPT ». Consulté le: 28 février 2025. [En ligne]. Disponible sur: <https://chatgpt.com/>
- [4] « THE 17 GOALS | Sustainable Development ». Consulté le: 27 avril 2025. [En ligne]. Disponible sur: <https://sdgs.un.org/goals>
- [5] « WAGO Download Center - WAGO Device Model - 751-9402 ». Consulté le: 27 avril 2025. [En ligne]. Disponible sur: <https://downloadcenter.wago.com/wago/null/details/m2dd83m229zdbc5cau>
- [6] « WAGO Download Center - WAGO Device Model - 751-9401 ». Consulté le: 27 avril 2025. [En ligne]. Disponible sur: <https://downloadcenter.wago.com/wago/null/details/m2ddbfiuhrn44rp2h>
- [7] « Accordion | React Bootstrap ». Consulté le: 27 avril 2025. [En ligne]. Disponible sur: <https://react-bootstrap.netlify.app/docs/components/accordion/>