Treball de Fi de Grau

Grau en Enginyeria en Tecnologies Industrials (GETI)

Desenvolupament d'un sistema de monitoreig i visualització per un generador Dish Stirling

MEMÒRIA

16 de juny de 2021

Autor: Víctor Ramos Medina

Director: Daniel Montesinos-Miracle

Convocatòria: 06/2021

Escola Tècnica Superior
d'Enginyeria Industrial de Barcelona

Resum

En aquest projecte s'ha realitzat un breu estudi d'un sistema de captació solar, anomenat dish Stirling. Concretament, s'ha estudiat el dish Stirling TRINUM que es troba instal·lat a la coberta de l'edifici de l'ETSEIB, UPC, Barcelona. Aquest aparell mancava de sistema de monitoratge, fet que dificultava la comprovació i el seguiment del seu funcionament. A més, l'absència d'un sistema de monitoratge no ha ajudat a evitar les múltiples fallades que el dish Striling ha patit durant els últims anys. L'any 2019 es va fer una proposta de sistema de monitoratge amb l'inconvenient que no era amb un software lliure.

Principalment, aquest treball consisteix en el desenvolupament d'un sistema informàtic mitjançant un software lliure que permeti el monitoratge i la visualització de dades elèctriques i tèrmiques obtingudes mitjançant el generador dish Stirling.

L'obtenció de les dades es realitza amb un sistema d'adquisició de dades ja existent i es podrà operar i visualitzar aquestes amb l'aplicació web desenvolupada. En aquest Treball Final de Grau s'ha creat el servidor web amb el software de programació Node-Red i s'han emmagatzemat les dades a una base de dades utilitzant el sistema de gestió MariaDB.

Índex

Índex de figures 4

Índex de taules 6

1 Introducció 7
- 1.1 Origen del projecte i motivació . . . 8
- 1.2 Requeriments previs . . . 8
- 1.3 Objectius del projecte . . . 8
- 1.4 Abast i limitacions del projecte . . . 9

2 Estat de l'art 10
- 2.1 Història del dish Stirling . . . 10
- 2.2 Components i funcionament del dish Stirling . . . 11
  - 2.2.1 Concentrador solar parabòlic . . . 11
  - 2.2.2 Sistema de generació. Motor Stirling . . . 14
  - 2.2.3 Receptor . . . 15
  - 2.2.4 Sistema de refrigeració . . . 16

3 Estudi d'alternatives 17

4 Recollida de dades del dish Striling TRINUM 19
- 4.1 Variables obtingudes . . . 20

5 Emmagatzematge de les variables. MariaDB 23
- 5.1 Llenguatge de programació. SQL . . . 23
- 5.2 Creació i manipulació de la base de dades . . . 24

6 Node-Red 29
- 6.1 Llenguatge de programació i tecnologies necessàries . . . 32
  - 6.1.1 JavaScript . . . 32
  - 6.1.2 Node.js . . . 33
  - 6.1.3 JSON . . . 34
- 6.2 Flow 1. Introducció dels valors de les variables a la base de dades . . . 35
- 6.3 Flow 2. Creació i personalització del Dashboard . . . 43
  - 6.3.1 Pantalla 1. Inici . . . 43
  - 6.3.2 Pantalla 2. Variables . . . 45
  - 6.3.3 Pantalla 3. Generació energètica . . . 49
  - 6.3.4 Pantalla 4. Consulta de variables . . . 53

7 Dashboard 57
- 7.1 Pantalla 1. Inici . . . 58
- 7.2 Pantalla 2. Variables . . . 61
- 7.3 Pantalla 3. Generació energètica . . . 63
- 7.4 Pantalla 4. Consulta de variables . . . 65

8 Planificació 67
- 8.1 Calendari . . . 67
- 8.2 Tasques del projecte . . . 68

8.3 Temps estimat i distribució temporal del projecte 69

9 Pressupost **71**
9.1 Cost de personal 71
9.2 Costos informàtics i d'equipament 72
9.3 Altres costos 73
9.4 Cost total 74

10 Impacte ambiental **75**

Conclusions **76**

Agraïments **78**

Referències **79**

Índex de figures

1 Consum mundial d'energia primària en 2018 [1] . . . 7
2 Procés que se seguirà fins a la consecució de l'objectiu principal [Font pròpia] . . 9
3 Representació d'un dish Stirling amb les seves parts [9] . . . 11
4 Divisió de la geometria d'un concentrador parabòlic en diverses parts [10] . . . 12
5 Representació dels eixos azimut i elevació [11] . . . 13
6 Tipus de motor Stirling [12] . . . 14
7 Cicle termodinàmic de Stirling [13] . . . 15
8 Arquitectura del sistema de monitorització del dish Stirling TRINUM [18] . . . 20
9 Visualització de les variables del TRINUM al servidor XML de l'EDS [Font Pròpia] 20
10 Procediment per la creació i utilització d'una base de dades en MariaDB [Font Pròpia] . . . 24
11 Procediment per la creació d'una taula dins de la base de dades [Font Pròpia] . . 25
12 Explicació de les comandes per crear la columna AMB_T [Font Pròpia] . . . 26
13 Descripció de la taula 'trinum' i taules existents a la base de dades [Font Pròpia] 26
14 Selecció de dades de la taula 'trinum' [Font Pròpia] . . . 27
15 Selecció de dades de la taula 'trinum' amb alguna condició [Font Pròpia] . . . 27
16 Interfície de programació de Node-Red [Font Pròpia] . . . 30
17 Exemple d'instal·lació dels nodes Dashboard de la llibreria Node-Red [Font Pròpia] 31
18 Configuració dels nodes principals de Node-Red [Font pròpia] . . . 32
19 Exemple de flow utilitzant els nodes principals de Node-Red [Font Pròpia] . . . 32
20 Nodes utilitzats (1) [Font pròpia] . . . 37
21 Nodes utilitzats (2) [Font pròpia] . . . 37
22 Configuració del node mysql [Font pròpia] . . . 38
23 Flow 1 a Node-Red [Font Pròpia] . . . 38
24 Configuració dels nodes change [Font pròpia] . . . 39
25 Configuració del node Inject [Font pròpia] . . . 40
26 Configuració del node http request [Font pròpia] . . . 40
27 Configuració del node XML [Font pròpia] . . . 41
28 Configuració del node Function [Font pròpia] . . . 41
29 Variables en format JSON [Font pròpia] . . . 42
30 Flow per l'elaboració de la pantalla 1 del Dashboard [Font pròpia] . . . 43
31 Configuració dels nodes change [Font pròpia] . . . 44
32 Nodes de la part 2 del flow de la pantalla 1 del dashboard [Font pròpia] . . . 44
33 Codi del node template [Font pròpia] . . . 45
34 Flow per l'elaboració de la pantalla 2 del Dashboard [Font pròpia] . . . 46
35 Contingut del node function anomenat "Mostrar grup" [Font pròpia] . . . 47
36 Contingut del node function anomenat "SQL" (Pantalla 2) [Font pròpia] . . . 47
37 Contingut del node function anomenat "Potència" [Font pròpia] . . . 48
38 Configuració del Node chart [Font pròpia] . . . 49
39 Flow per l'elaboració de la pantalla 3 del Dashboard [Font pròpia] . . . 50
40 Contingut del node function anomenat "SQL diari" [Font pròpia] . . . 51
41 Contingut del node function anomenat "Diari elec" [Font pròpia] . . . 52
42 Contingut del node function anomenat "Text diari" [Font pròpia] . . . 53
43 Flow per l'elaboració de la pantalla 4 del Dashboard [Font pròpia] . . . 54
44 Configuració del Node text input [Font pròpia] . . . 54
45 Contingut del node function anomenat "SQL" (Pantalla 4) [Font pròpia] . . . 55
46 Configuració dels nodes text i gauge (Pantalla 4) [Font pròpia] . . . 56

47 Configuració general del dashboard [Font pròpia] 57
48 Layout editor de la pantalla del dashboard "Generació energètica" [Font pròpia] 58
49 Configuració general del dashboard [Font pròpia] 58
50 Mock-Up de la pantalla 1 del dashboard [Font pròpia] 59
51 Resultat final de la pantalla 1 del dashboard [Font pròpia] 60
52 Menú lateral del dashboard [Font pròpia] 60
53 Mock-Up de la pantalla 2 del dashboard [Font pròpia] 61
54 Resultat final de la pantalla 2 del dashboard (1) [Font pròpia] 62
55 Aspecte dels filtres de la pantalla 2 del dashboard [Font pròpia] 62
56 Resultat final de la pantalla 2 del dashboard (2) [Font pròpia] 63
57 Mock-Up de la pantalla 3 del dashboard [Font pròpia] 63
58 Resultat final de la pantalla 3 del dashboard [Font pròpia] 64
59 Mock-Up de la pantalla 4 del dashboard [Font pròpia] 65
60 Resultat final de la pantalla 4 del dashboard (1) [Font pròpia] 66
61 Resultat final de la pantalla 4 del dashboard (2) [Font pròpia] 66
62 Calendari del TFG per la convocatoria del quadrimestre de primavera [32] 67
63 Diagrama de Gannt del projecte [Font Pròpia] 70

Índex de taules

1 Taula de reflectivitat i emissivitat de materials [9] 13
2 Taula amb les alternatives i la decisió final [Font pròpia] 18
3 Taula de les variables obtingudes i el seu corresponent identificador [Font pròpia] 21
4 Taula de les variables calculades i el seu corresponent identificador [Font pròpia] 22
5 Taula amb la dedicació en hores de les diferents tasques [Font pròpia] 69
6 Taula amb el cost degut a les hores de cada treballador [Font pròpia] 71
7 Taula amb el cost degut a les hores dedicades a cada tasca del projecte [Font pròpia] 72
8 Taula amb el costos informàtics i d'equipament [Font pròpia] 73
9 Taula amb el cost energètic i de connexió a Internet [Font pròpia] 74
10 Taula amb el cost total del projecte [Font pròpia] 74

1 Introducció

Durant els últims anys el consum energètic ha tingut un creixement exponencial a causa de diversos factors com el creixement demogràfic o els avanços tecnològics. Com es pot veure al gràfic inferior (Fig. 1), l'any 2018 només el 10,8% del consum mundial d'energia va ser d'energies renovables. El canvi climàtic patit els últims anys no ha fet més que accelerar la necessitat d'impulsar les energies renovables en detriment de l'ús de combustibles fòssils. Les energies renovables es caracteritzen per provenir de fonts naturals inesgotables, com l'energia solar o l'energia eòlica, o per ser capaces de regenerar-se en poc temps, com la biomassa. Per tant, és evident que per tal de mantenir de manera sostenible el consum energètic s'ha d'augmentar la contribució de les energies renovables en aquest consum.

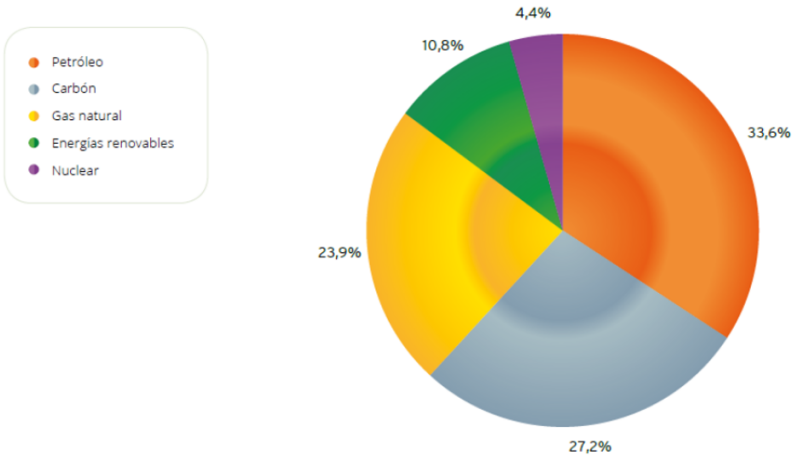


Figura 1: Consum mundial d'energia primària en 2018 [1]

Per tal de promoure el consum d'energies renovables, l'any 2015 la Universitat Autònoma de Barcelona va contactar amb el centre *CITCEA-UPC* per formar part d'un projecte estratègic que complís amb aquests objectius. Aquest projecte, anomenat *DIDSOLIT* (Development and Implementation of Decentralised Solar-energy-related Innovative Technologies) i finançat per la Unió Europea, va promoure la transferència de coneixements en el camp de la generació elèctrica de font solar descentralitzada i l'aplicació de tecnologies innovadores d'escala reduïda que poden ser integrades en edificis o dependències públiques [2]. A conseqüència d'aquest projecte, es va instal·lar un nou sistema de captació solar anomenat dish Stirling a la coberta de l'edifici de l'ETSEIB, UPC, Barcelona. Aquest equipament va ser adquirit a l'empresa Innova Solar i es caracteritza per ser una font d'energia renovable, concretament d'energia solar amb un impacte ambiental mínim. A més, el dish Stirling pot generar 1kW d'electricitat i 3kW d'aigua calenta. L'electricitat produïda ha servit per a l'autoconsum, i l'aigua calenta per a les dutxes del gimnàs de la universitat [3].

Per tal de garantir un bon funcionament del dish Stirling de l'ETSEIB és imprescindible tenir un bon sistema de monitoratge i visualització de les dades que s'obtenen amb el captador solar. Amb aquest sistema es podria detectar qualsevol fallida del dish Stirling i es podria comprovar que el seu funcionament és el correcte. La creació del sistema de monitoratge i visualització de dades és la base de la realització d'aquest treball.

1.1 Origen del projecte i motivació

La realització d'aquest projecte neix de l'interès en el sector de la informàtica i de les energies renovables. Aquests dos camps estaran involucrats en aquest treball, ja que, com s'ha comentat anteriorment, es crearà un sistema per monitoritzar i visualitzar les dades obtingudes per un Dish Stirling, un aparell que permet generar energia gràcies a la radiació solar. L'interès per aquests dos sectors el tenia des d'abans d'iniciar el Grau d'Enginyeria en Tecnologies Industrials, però, durant la realització d'aquest grau, algunes de les assignatures cursades no han fet més que augmentar la curiositat per aquests dos àmbits.

L'elaboració del treball pot ser una bona oportunitat per millorar i obtenir nous coneixements sobre programació de base de dades i sobre la creació de dashboards. A més, es podrà conèixer com es realitza l'obtenció de dades energètiques d'un sistema Dish Stirling.

1.2 Requeriments previs

Per poder dur a terme aquest projecte s'utilitzarà l'eina de programació Node-Red i el sistema de gestió de base de dades MariaDB. Per tant, serà necessari tenir prèviament uns coneixements bàsics de comunicacions, d'informàtica i de manipulació de base de dades. Concretament, s'utilitzaran els llenguatges de programació JavaScript i Sql, per la qual cosa seria convenient tenir experiència en aquests dos llenguatges de programació o, almenys, en llenguatges de programació similars que facilitin l'aprenentatge d'aquests dos.

També serà imprescindible una bona capacitat d'aprenentatge per tal de poder adquirir el coneixement necessari i el temps i la capacitat d'esforç suficient per finalitzar el treball de manera satisfactòria en el termini establert.

1.3 Objectius del projecte

La realització d'aquest projecte persegueix el compliment dels següents objectius:

- Estudi i investigació del funcionament de la tecnologia Stirling. Principalment, s'estudiarà les parts que componen el dish Stirling i els principis termodinàmics que permeten la generació de l'energia.
- Estudi del sistema dish Stirling instal·lat a l'ETSEIB.
- Anàlisi i enteniment del sistema d'adquisició de dades inclòs en el dish Stirling TRINUM de l'ETSEIB. S'analitzarà com són recollides les dades i com seran emmagatzemades posteriorment a la seva obtenció.
- Creació d'un sistema de monitoratge i visualització d'aquestes dades aportades pel dish Stirling mitjançant el software Node-RED. Aquest software haurà de permetre presentar les dades de manera entenedora i l'usuari haurà de poder seleccionar les dades que vol que apareguin als gràfics. Aquest és l'objectiu principal del projecte.
- Anàlisi de les dades i els resultats obtinguts. Amb les dades recollides es durà a terme el càlcul de rendiments, potències, energies i altres característiques.

A la imatge següent (Fig.2) es pot observar el procés que se seguirà des de l'obtenció de les dades del dish Stirling TRINUM fins a aconseguir l'objectiu principal, la creació del dashboard.

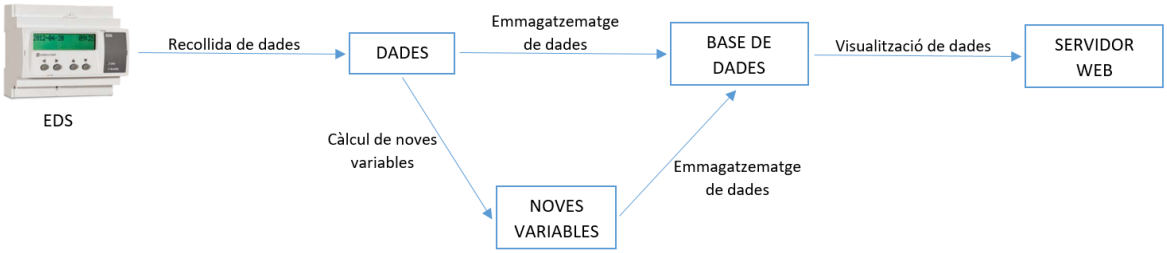


Figura 2: Procés que se seguirà fins a la consecució de l'objectiu principal [Font pròpia]

1.4 Abast i limitacions del projecte

Un cop descrits els objectius del projecte es pot definir l'abast d'aquest. El compliment de l'objectiu principal, és a dir, la creació d'un servidor web que permeti el monitoratge i la visualització de les dades del dish Stirling, permetrà la consulta de l'estat de diverses variables tèrmiques i elèctriques per part de l'usuari. A més, es podrà fer una consulta de l'històric d'aquestes dades per tal de poder veure la seva evolució en el temps. La consulta de les dades recollides permetrà comprovar si el funcionament del dish Stirling és correcte i així poder detectar possibles errors en el futur.

Per tal d'aconseguir l'abast desitjat serà imprescindible conèixer les possibles limitacions i com superar-les. Entre aquestes limitacions podem trobar:

- Limitació temporal. Com a qualsevol projecte, el TFG té unes dates d'entrega establertes i, per tant, s'haurà de fer el treball en un termini establert. Per això, s'haurà de treballar de manera constant per poder assolir els objectius definits.

- Limitació de pressupost. A l'hora d'escollir el software per dur a terme el projecte, s'haurà de tenir en compte la limitació de pressupost existent. És per això que s'ha decidit realitzar el treball amb el software de llicència lliure Node-RED.

- Limitació de coneixements. Per a una òptima realització del projecte s'haurà de tenir coneixements en l'àmbit de la informàtica. Per tant, serà necessari un bon aprenentatge previ per assolir les competències necessàries per programar, crear dashboards i emmagatzemar variables en base de dades.

- Altre factor que pot limitar l'assoliment dels objectius serà l'aparició d'errors. S'haurà d'evitar errors tant en la programació com en l'emmagatzematge de dades i s'haurà de comprovar que la recollida de dades és correcta. L'aparició d'un error podria provocar futurs problemes que dificultarien la finalització del projecte de manera satisfactòria.

2 Estat de l'art

Abans de començar a treballar amb el sistema és imprescindible contextualitzar i conèixer el funcionament de dish Stirling. També és important entendre el context en el que es va crear i quin objectiu perseguia la seva creació inicialment.

2.1 Història del dish Stirling

Durant els primers anys del segle XIX el motor més utilitzat era la màquina de vapor. Aquesta provocava nombrosos accidents deguts a l'explosió de la màquina a causa de les altes pressions de treball. Aquests accidents implicaven grans costos als propietaris de les màquines i inclús provocava la mort dels operaris que treballaven amb la màquina de vapor. Amb la motivació d'evitar aquests accidents, Robert Stirling, un clergue i inventor escocès, va crear el motor Stirling. Concretament, va demanar la patent l'any 1816 i el motor Stirling es va convertir en el primer motor d'aire calent en funcionament. [4]

Durant els següents anys, l'eficiència del motor va anar millorant gràcies a la invenció del regenerador (anomenat economitzador en aquell moment), un element que com s'explicarà més endavant a l'apartat 2.2.2 permet emmagatzemar la calor d'un cicle termodinàmic quan el gas disminueix de temperatura a volum constant i subministra aquesta calor quan el gas torna a augmentar la temperatura.

Tot i el bon rendiment que donava el motor Stirling va deixar d'usar-se gràcies a la creació de motors amb millor eficiència com el motor Otto o Diesel. No va ser fins a l'any 1950 quan el motor va tornar a ser molt utilitzat gràcies al fet que la companyia holandesa Philips va crear un generador elèctric basat en la tecnologia Stirling. Una de les aplicacions del motor Stirling és la seva utilització en tecnologies de dish Stirling, on s'aconsegueix el focus de temperatura calenta a partir d'un disc parabòlic que concentra la radiació solar en un punt, obtenint el focus calent del motor. Un dels primers sistemes dish Stirling va ser l'anomenat Vanguard instal·lat a Califòrnia l'any 1983, el MDAC-25 disc a Los Angeles l'any 1984 i els discos Schlaich, Bergermann and Partner (SBP) instal·lats l'any 1984 a l'Aràbia Saudita. El disc *Vanguard* va establir un rècord d'eficiència entre radiació solar directe i conversió elèctrica d'un 30%.

Actualment, els sistemes dish Striling no són tan utilitzats com altres tecnologies solars, però el constant avenç en aquesta tecnologia fa preveure una disminució del seu cost que pugui impulsar la seva utilització. Tot i que els sistemes dish Stirling tenen un alt coeficient de conversió d'energia solar a elèctrica (aproximadament del 30%), la seva potència unitària és baixa (inferior a 25 kW) i això obstaculitza la seva utilització a gran escala. Per tant, la seva aplicació més habitual és en habitatges particulars, encara que aquesta tecnologia té la capacitat d'operar tan individualment per aplicacions remotes com de manera agrupada en forma de granges i connectant-se a la xarxa. [5] [6]

La gran eficiència i les diverses aplicacions de la tecnologia dish Stirling fan que el seu ús estigui en un creixement continu i és previsible que en el futur sigui una tecnologia molt més accessible i rendible econòmicament i amb un millor rendiment. Un dels últims projectes que més ha impulsat aquesta tecnologia ha sigut el projecte *Stirling Energy System of Arizona* [7] de l'any 2006 on es va aprovar la instal·lació de 12000 discos reflectors al desert d'Imperial Valley per produir 300 MW.

2.2 Components i funcionament del dish Stirling

Els sistemes dish Striling o sistemes de disc parabòlic (SDP) són equips que transformen l'energia provinent de la radiació solar en energia elèctrica i tèrmica (en cas que sigui un sistema amb cogeneració). Aquests sistemes estan formats per un gran disc reflector parabòlic acoblat a un motor Stirling que s'ubica a la zona focal del concentrador per transformar la radiació solar en electricitat. A més, el disc està format per un conjunt de miralls que reflecteixen i concentren la radiació solar en un receptor. D'aquesta manera s'assoleix una temperatura molt elevada que provoca que es pugui convertir la calor en treball eficientment. Per tal de poder absorbir la màxima radiació solar possible els sistemes dish Stirling incorporen un mecanisme format per un eix azimut que permet seguir la trajectòria solar. [8]

Els dish Stirling, per la seva grandària, independència i modularitat, tenen la capacitat d'abastir d'electricitat a regions on la densitat de població és baixa i dispersa, fet que amb sistemes convencionals resulta poc rendible.

A la imatge inferior (Fig. 3) es pot observar un esquema d'un dish Stirling on s'indiquen les parts que el componen i que s'explicaran a continuació.

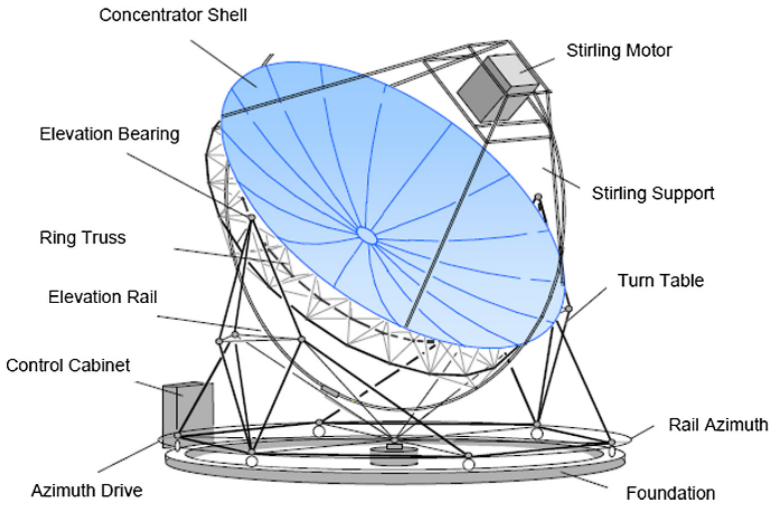


Figura 3: Representació d'un dish Stirling amb les seves parts [9]

2.2.1 Concentrador solar parabòlic

Els sistemes dish Stirling estan formats per un gran mirall parabòlic anomenat concentrador solar parabòlic, que enfoca els rajos solars cap al costat calent d'un motor Stirling. L'objectiu principal del concentrador solar és concentrar la màxima radiació solar que sigui possible en una àrea reduïda (anomenada focus) on està situat el receptor. És per això que té forma parabòlica i està fabricat de material reflectant.

A l'hora de fabricar el disc parabòlic s'ha de tenir en compte les propietats que haurà de tindre: pes raonable, duresa contra la deflexió i la càrrega del vent, durabilitat contra la humitat i els canvis de temperatura (haurà de suportar les temperatures mínimes i màximes de la zona geogràfica on s'instal·li), les peces han de ser flexibles, materials reflectants eficaços i de baix cost i vida útil llarga.

La grandària del concentrador ve determinada per la potència nominal i l'energia generada en un cert període de temps i en unes determinades condicions de radiació solar i rendiment del sistema. La radiació solar màxima que es sol rebre és de 1000 W/m^2. Per tant, la mida del diàmetre dels discs dependrà de la quantitat d'energia a produir i oscil·larà entre valors des de 3 metres fins a 20 metres.

Com s'ha explicat anteriorment, el concentrador solar té forma de paràbola de revolució, ja que aquesta geometria maximitza la reflexió de la radiació solar cap al focus i, d'aquesta manera, es poden assolir temperatures de treball molt elevades. El fet d'assolir aquestes altes temperatures és imprescindible per aconseguir un bon rendiment del cicle termodinàmic.

La temperatura màxima que pot assolir un cos en el seu punt focal es dóna suposant que el receptor és un cos negre, és a dir, un cos que absorbeix tota l'energia radiant incident des de qualsevol direcció i per a totes les longituds d'ona, sense reflectir-la, ni transmetre-la, ni dispersar-la. Aquesta temperatura màxima que es pot assolir ve definida per la següent expressió:

$$C_G I_S = \epsilon\sigma \, \mathrm{T}^4$$

On $C_G I_S$ és el flux d'energia lumínica i $\epsilon\sigma \, \mathrm{T}^4$ és el flux tèrmic. No obstant això, com que el disc no serà totalment un cos negre i tindrà pèrdues de conducció, convecció i radiació no es pot assolir aquesta temperatura màxima ideal.

Com s'ha explicat anteriorment, la geometria d'un paraboloide és la més adient per concentrar la radiació solar en un punt. A la pràctica, resulta complicat fabricar un paraboloide perfecte, de manera que normalment es realitzen aproximacions tal que els costos siguin raonables, usualment dividint la geometria en peces més simples, com es pot observar a la imatge següent (Fig.4):

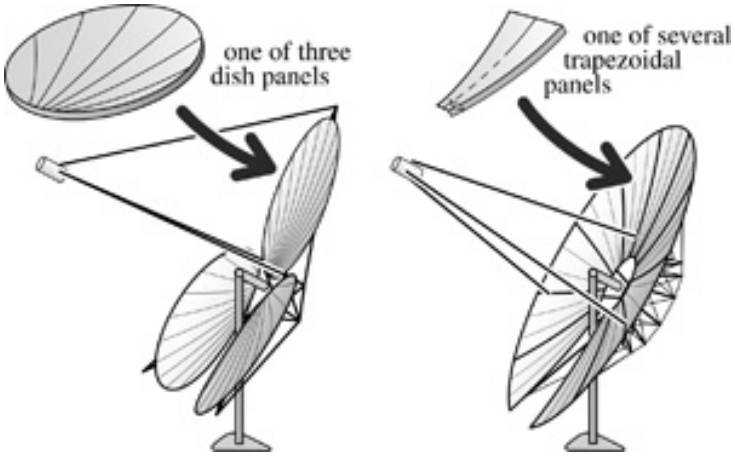


Figura 4: Divisió de la geometria d'un concentrador parabòlic en diverses parts [10]

La selecció del material del concentrador és un dels factors més importants a tenir en compte a l'hora de dissenyar i fabricar el plat parabòlic. La reflectivitat del material del concentrador afecta el percentatge de radiació solar que arriba al receptor: com més gran sigui la reflectivitat major serà la radiació que li arriba. La següent taula 1 mostra la reflectivitat d'alguns materials que es poden utilitzar en el disseny del concentrador de plaques solars.

Materials	Reflective (%)	Emissive (%)
Polymeric film, non metal	98	2
Aluminum, acrylic	98	2
Silver, aluminum acrylic	97	3
Silver, acrylic	95	5
Aluminum	86	14
Aluminum, polyethylene	97	3
Plexiglas with mirror	90	10
Thermoplastic, silver, gold, brass, etc.	80	20
Aluminum mylar	97	3
Polymer, copper, silvered, alumina	97	3
Polished stainless	50	50
Ceramic metallic coating layer	95	5
Glass/silver 4 mm	93.8	6.2
Glass/silver 2 mm	94	6
Glass/silver 1 mm	94.6	5.4
Miro 2–95	88.6	11.4
Miro 3–95	91.1	8.9
Anod aluminum	86.8	13.2
1000.90	89.8	10.2
ECP305+/aluminum	95.6	4.4
ECP305+/glass	96.1	3.9
Sunflex (polymer/aluminum)	86.9	10.1
SA 85/glass	88.1	11.9
SA 85/steel	88.2	11.8
Sol-gel coated silver	95.5	4.5
Sol-gel coated aluminum	91	9

Taula 1: Taula de reflectivitat i emissivitat de materials [9]

El material més emprat per la fabricació de concentradors solars parabòlics és l'alumini. Tot i no ser el material amb la reflectància més elevada dels que apareixen a la taula, el seu baix cost i les seves propietats de densitat i resistència a canvis atmosfèrics el fan un material adient en la fabricació d'aquests sistemes. També és habitual formar els concentradors parabòlics amb segments de resina de fibra de vidre.

Seguiment solar

Per tal de maximitzar la radiació solar rebuda, el dish Stirling incorpora un sistema de seguiment solar. Aquest sistema es realitza mitjançant dos eixos: l'azimut i l'elevació, que es troben representats a la imatge inferior (Fig. 5). L'azimut permet que el dish giri amb un eix vertical (perpendicular a la Terra) i l'elevació que permet al dish girar amb un eix perpendicular a l'eix azimut.

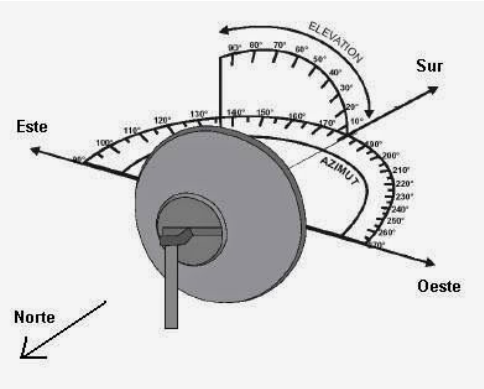


Figura 5: Representació dels eixos azimut i elevació [11]

2.2.2 Sistema de generació. Motor Stirling

El sistema de generació d'un dish Stirling consta d'un motor Stirling i d'un alternador. Tal com es va explicar a l'apartat 2.1, el motor Stirling s'encarrega de transformar la calor rebuda en energia mecànica. El principi de funcionament és el treball realitzat per l'expansió i contracció d'un gas (que pot ser heli, hidrogen, nitrogen o aire) el qual segueix un cicle de refredament en un focus de fred, on es contrau, i d'escalfament en un focus calent, on s'expandeix. En el cas del dish Stirling, el focus calent s'aconsegueix mitjançant la calor rebuda provinent de la radiació solar. Per tant, és necessari que existeixi un gradient de temperatura entre els dos focus i per això el motor Stirling és considerat un motor tèrmic.

El gran avantatge d'aquest motor és la seva versatilitat en la utilització de fonts d'energia pel seu funcionament, ja que només necessita una font de calor externa al motor, fent possible emprar una gran varietat de fonts energètiques (energia solar tèrmica, combustibles, ús de la biomassa, energia geotèrmica). A més, la baixa probabilitat de patir explosions el fan un motor molt segur i amb una vida útil molt llarga. També té un rendiment molt alt que és superior al rendiment de la majoria de motors convencionals.

En funció de la configuració del motor Stirling hi ha diferents tipus. Els principals són el tipus alpha, tipus beta i tipus gamma. A la imatge següent (Fig. 6) es pot observar un esquema d'aquests tres tipus de motor Stirling.

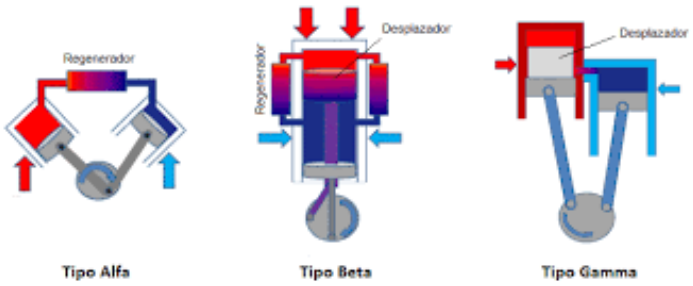


Figura 6: Tipus de motor Stirling [12]

El cicle termodinàmic que segueix el motor Stirling és el cicle de Stirling i és molt similar al cicle ideal de Carnot. Aquest cicle consta de quatre processos que es poden veure a la imatge inferior (Fig. 7):

- **1-2**. Expansió isotèrmica (a temperatura constant) del fluid de treball a alta temperatura. Durant el procés s'absorbeix calor de la font de temperatura calenta.
- **2-3**. Cessió de calor isocora (a volum constant) del fluid de treball reduint la seva temperatura.
- **3-4**. Compressió isotèrmica (a temperatura constant) del fluid de treball a la temperatura inferior. Se cedeix calor a la font freda.
- **4-1**. Escalfament isocor (a volum constant) del fluid de treball amb absorció de calor. La temperatura augmenta fins a la temperatura inicial.

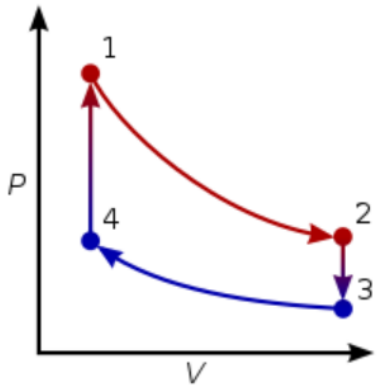


Figura 7: Cicle termodinàmic de Stirling [13]

Teòricament el rendiment del motor Stirling és igual al rendiment de Carnot, que és el rendiment màxim d'una màquina tèrmica (rendiment termodinàmic ideal). Tot i això, a causa de pèrdues aquest valor de rendiment no es pot aconseguir.

Regenerador

Un dels elements principals dels motors Stirling és el regenerador. El regenerador és un intercanviador de calor intern que absorbeix i cedeix calor a les etapes a volum constant del cicle termodinàmic. Aquest element emmagatzema la calor cedida al procés **2-3** i després el subministra al procés **4-1**.

El regenerador consisteix en un medi porós amb conductivitat tèrmica menyspreable, que conté un fluid. Aquest element s'utilitza per dividir el motor en dues zones: una zona calenta i una altra zona freda. El fluid es desplaça de la zona calenta a la freda durant els diversos cicles de treball, travessant el regenerador.

2.2.3 Receptor

El dish Stirling inclou un element que s'encarrega de l'absorció de la radiació solar reflectida pel concentrador parabòlic, aquest element és el receptor. Aquesta energia absorbida la transfereix al fluid de treball que utilitza el motor.

Existeixen diversos tipus de receptors d'entre els quals destaquen: els receptors de cavitat i els receptors externs. En els receptors de cavitat, la radiació entra per una obertura situada en el focus del paraboloide i posteriorment incideix en l'absorbidor. Els receptors externs són esfèrics i per tant poden rebre la radiació solar des de qualsevol direcció, això fa que la seva eficiència sigui més gran, però també té pèrdues tèrmiques majors que els receptors de cavitat. Els receptors que s'utilitzen als dish Stirling són els receptors de cavitat, ja que les pèrdues per radiació i convecció són menors.

Un dels elements principals del receptor és l'absorbidor que té la funció de transmetre l'energia que capta el receptor al fluid de treball. Aquest absorbidor pot ser de tres tipus: de tubs directament il·luminats, de reflux o volumètrics pressuritzats. Tot i això, als dish Stirlings només s'utilitzen els absorbidors de tubs directament il·luminats, que consisteixen en un feix de tubs per un circula el fluid de treball i sobre els quals incideix directament la radiació solar. [14]

2.2.4 Sistema de refrigeració

Per garantir el bon funcionament del sistema, és important eliminar la calor del motor mitjançant un fluid refrigerant. Aquest fluid sol ser aigua barrejada amb anticongelant, que sovint és Propilenglicol. El refrigerador és l'encarregat de mantenir la temperatura del focus fred dintre de l'interval adequat per assegurar que no es produeix cap problema.

En cas que s'utilitzi un sistema amb cogeneració, la calor que rep del motor el líquid refrigerant pot ser emmagatzemada com energia tèrmica i ser aprofitada en altre moment. En cas que no es tingui cogeneració, la calor del fluid de treball es perd amb l'ambient. Habitualment, s'utilitza una bomba per assegurar una bona circulació del fluid refrigerant pel circuit tancat.

3 Estudi d'alternatives

Un cop definits els objectius del projecte i després d'haver estudiat i entès el funcionament del sistema dish Stirling s'ha hagut de decidir com s'executarà el projecte. Principalment, s'han de prendre decisions sobre com emmagatzemar les variables en una base de dades, quina base de dades utilitzar i amb quina eina fer el dashboard.

Primerament, per introduir els valor de les variables a la base de dades s'ha decidit utilitzar una eina anomenada Node-Red, proposada pel tutor del treball al principi d'aquest. Aquesta eina compleix el requisit fonamental que sigui un software lliure i permet fer aquesta tasca sense cap problema. Per aquest motiu, seguint amb les recomanacions del tutor, s'utilitzarà Node-Red per afegir les variables del dish Stirling a la base de dades. A més, per escollir la base de dades s'ha de tenir en compte que Node-Red pot treballar principalment amb dues bases de dades: MySQL i SQLite. Per decidir quina d'aquestes dues bases de dades emprar s'analitzaran les prestacions de cadascuna per valorar els seus avantatges i inconvenients [15] [16].

- **Suport de tipus de dades.** SQLite només accepta 5 tipus de dades: Blob (binari), Integer (nombre enter), Null (nul), Text i Real (nombre real). MySQL en canvi, a més d'aquest tipus de dades, accepta fins a 25 tipus més de dades. Per tant, MySQL aporta molta més flexibilitat pel que fa al tractament de dades.

- **Emmagatzematge i portabilitat.** SQLite té una biblioteca d'una mida d'aproximadament 250 KB i emmagatzema en un sol arxiu tota la informació. Per contra, MySQL té una biblioteca d'uns 600 MB i emmagatzema la informació en diferents arxius, els quals s'han de condensar en un sol abans de copiar o exportar MySQL.

- **Accés múltiple i escalabilitat.** SQLite no té la possibilitat d'accés de múltiples usuaris mentre que MySQL compta amb un sistema d'administració d'usuaris que permet l'accés de múltiples usuaris als quals es pot atorgar diferents nivells de permís. Pel que fa a l'escalabilitat (capacitat de variar el seu comportament en funció dels requeriments), amb SQLite el requisit de memòria creix quan augmenta la base de dades, per tant és adequat en bases de dades petites. MySQL té una escalabilitat major i permet tenir bases de dades grans amb un major rendiment.

- **Seguretat i facilitat d'instal·lació.** En el cas de SQLite, no fa falta configurar-lo ni instal·lar-lo, és suficient amb descarregar les biblioteques a l'ordinador. Per altra banda, MySQl sí que necessita configurar-se, però no representa molta dificultat. Pel que fa a la seguretat, SQLite no té cap mètode d'autenticació i, per tant, qualsevol pot accedir a les dades. MySQL en canvi, inclou una autenticació amb un nom d'usuari i contrasenya. Això fa que MySQL sigui una base de dades molt més segura.

Vist el comportament d'ambdues bases de dades enfront de diferents característiques, es pot concloure en quines situacions seria adient utilitzar cadascuna d'elles. SQLite és recomanable per gestionar aplicacions portables i que no necessitin emmagatzemar moltes dades. En canvi, MySQL és una millor opció quan s'ha de fer la gestió d'un sistema que requereix seguretat i que necessita emmagatzemar un gran nombre de dades. Per tant, tot i que ambdues bases de dades es podrien haver adaptat als requeriments, s'ha escollit MySQL per la seva possibilitat d'emmagatzemar més dades amb un millor rendiment. Concretament, s'ha utilitzat la base de dades MariaDB, que és una versió millorada de MySQL.

Finalment, queda decidir quina eina utilitzar per fer el dashboard. Donat que la principal funció del servidor web serà mostrar gràfics de les variables del dish Stirling, s'ha de buscar un software que sigui idoni per fer aquest tipus de dashboard. La primera opció que va sorgir va ser utilitzar Grafana, que és un programari lliure que permet la visualització de dades de sèrie temporal mitjançant la creació de quadres de comandament i gràfics. Per tant, és un software que encaixa perfectament amb l'eina buscada. Tot i això, a mesura que s'ha anat aprenent com utilitzar Node-Red (el software utilitzat per introduir els valors a la base de dades) s'ha vist que aquest software també té la capacitat de fer dashboards amb les característiques requerides. Per aquest motiu, es va decidir utilitzar el Node-Red per ambdues tasques, ja que això permetrà reduir el temps que s'hagués d'haver dedicat a l'aprenentatge del funcionament de Grafana.

Es pot veure un resum de les alternatives presentades i de la decisió final presa a la taula següent (2):

ELEMENT	ALTERNATIVES	SELECCIÓ FINAL
Software per introduir les variables a la base de dades	Node-Red	Node-Red
Base de dades	MySQL SQLite	MySQL (MariaDB)
Software per fer el dashboard	Node-Red Grafana	Node-Red

Taula 2: Taula amb les alternatives i la decisió final [Font pròpia]

4 Recollida de dades del dish Striling TRINUM

Un cop explicat el funcionament d'un sistema dish Stirling s'entrarà en detall en com es realitza la recollida de dades del dish Stirling TRINUM de la coberta de l'edifici de l'ETSEIB. El sistema incorporat en el TRINUM per la recollida de dades és imprescindible per fer un seguiment de l'estat del sistema i assegurar un correcte funcionament de les operacions. A més, la recollida de dades permetrà la posterior visualització de les variables recopilades amb sensors instal·lats al sistema.

L'extracció de les dades es duu a terme amb un software anomenat Power Studio Scada. Aquest software s'utilitza per gestionar equips de control energètic i permet realitzar estudis energètics i conèixer consums energètics per unitat produïda. Les seves característiques principals són la configuració de dispositius i la creació de pantalles i informes amb els quals visualitzar el valor actual de les variables i la seva evolució en el temps. Per poder extreure les dades amb el software Power Studio es necessiten 3 elements que, units com s'observa a la imatge inferior (Fig. 8), permet la recollida dels valors de les variables de manera satisfactòria. A continuació s'explicaran els elements necessaris per a la correcta recollida de les dades:

- **EDS (Efficiency Data Server)**. És un gestor d'eficiència energètica. L'equip està proveït d'un total de 8 entrades i 6 sortides digitals a través d'un relé i disposa d'un bus de comunicació RS-485 (un camí de comunicació entre dos o més dispositius que permet transmetre informació) que li permet comunicar amb equips de camp externs. A més, l'EDS representa i emmagatzema la informació a través de la seva connexió Ethernet i el seu servidor Web integrat. També disposa d'un servidor XML estàndard, a través del qual altres aplicacions externes poden integrar de forma fàcil i intuïtiva la informació procedent del dispositiu. El sistema té la IP 147.83.75.6 i al present treball s'extraurà el valor de les variables accedint a l'enllaç: `http://147.83.75.6/services/user/values.xml?id=Trinum?`. A més, es pot accedir a diversos enllaços per trobar informació sobre les variables, sobre el dispositiu i sobre com canviar la configuració general. [17]

- **TCP1RS+**. És una passarel·la orientada a la conversió del medi físic Ethernet a RS-485 i viceversa. Aquesta producció es produeix amb el protocol Modbus/TCP. L'equip és pot parametritzar completament, podent configurar qualsevol paràmetre relatiu al port de comunicació Ethernet.

- **RS2RS**. És un convertidor que permet connectar a una xarxa RS-485 equips amb comunicacions RS-232. També pot funcionar com a repetidor o amplificador d'una xarxa RS-485. La utilització de l'RS2RS com a amplificador, permet augmentar la longitud del bus RS-485 d'una instal·lació i es pot utilitzar en instal·lacions en què apareguin problemes de comunicacions a causa d'interferències o per culpa de les distàncies del cablejat. A més, el convertidor és molt utilitzat en l'àmbit industrial, ja que consta d'un aïllament galvànic.

Com s'observa a la imatge (Fig. 8), el TRINUM dóna la informació en un bus RS-232 que es converteix en un bus RS-485 amb un dispositiu RS2RS. Posteriorment, el bus RS-485 es converteix amb un bus TCP/IP amb l'element TCP1RS+ i s'envia al software Power Studio. Finalment, la informació arriba a l'EDS.

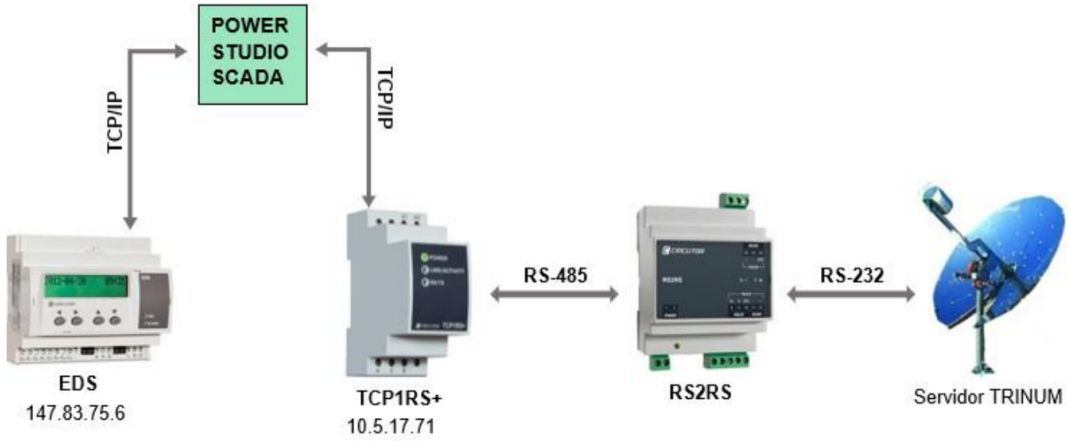
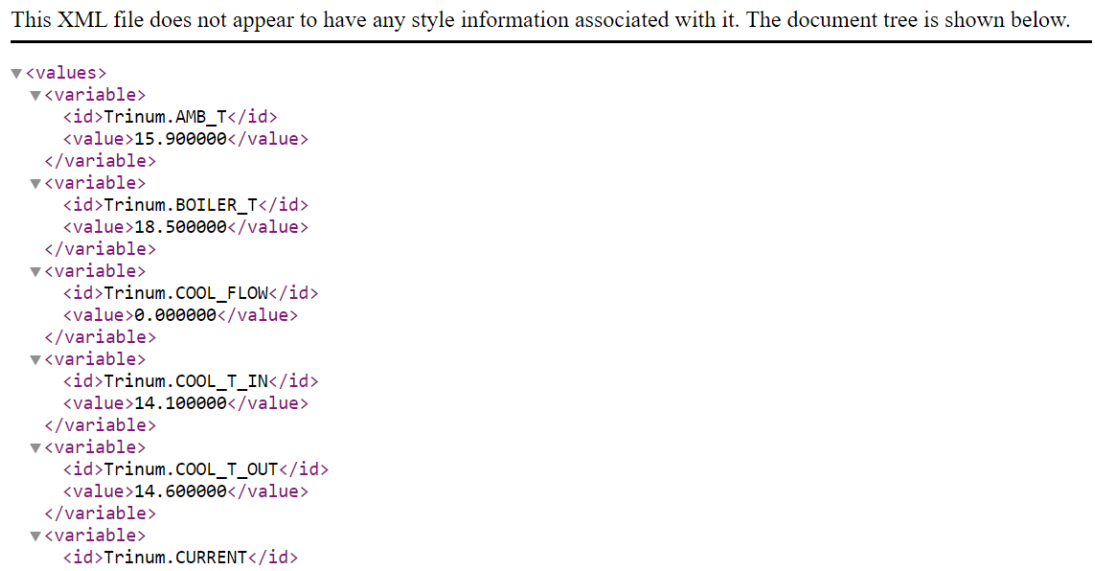


Figura 8: Arquitectura del sistema de monitorització del dish Stirling TRINUM [18]

El problema principal d'aquest sistema de recollida de dades és l'alt cost del software Power Studio Scada (d'aproximadament 185 €/mes en cas de contractar la llicència d'un any). És per aquest motiu que s'ha desenvolupat aquest projecte, ja que es vol crear un servidor que pugui complir amb la tasca que realitzaba el software Power Studio però amb un software lliure, sense cap cost. D'aquesta manera, el procediment que se seguirà, com es pot veure en els capítols que hi ha a continuació, consisteix a obtenir les dades del servidor XML de l'EDS, emmagatzemar-les en una base de dades i mostrar-les en un servidor web. Tot això es farà amb el software lliure Node-Red i s'instal·larà en un servidor Linux més endavant.

4.1 Variables obtingudes

Com s'ha comentat anteriorment, s'obtindrà el valor de les variables del dish Stirling TRINUM del servidor XML de l'EDS (Fig. 9).


This XML file does not appear to have any style information associated with it. The document tree is shown below.

```
▼<values>
  ▼<variable>
     <id>Trinum.AMB_T</id>
     <value>15.900000</value>
   </variable>
  ▼<variable>
     <id>Trinum.BOILER_T</id>
     <value>18.500000</value>
   </variable>
  ▼<variable>
     <id>Trinum.COOL_FLOW</id>
     <value>0.000000</value>
   </variable>
  ▼<variable>
     <id>Trinum.COOL_T_IN</id>
     <value>14.100000</value>
   </variable>
  ▼<variable>
     <id>Trinum.COOL_T_OUT</id>
     <value>14.600000</value>
   </variable>
  ▼<variable>
     <id>Trinum.CURRENT</id>
```

Figura 9: Visualització de les variables del TRINUM al servidor XML de l'EDS [Font Pròpia]

A continuació es mostrarà una taula amb les variables del sistema que s'emmagatzemaran a la base de dades i el seu identificador a l'XML (Taula 3).

IDENTIFICADOR	VARIABLE
AMB_T	Temperatura ambient [ºC]
BOILER_T	Temperatura acumulador[ºC]
COOL_FLOW	Cabal fluid refrigerant [l/min]
COOL_T_IN	Temperatura entrada motor [ºC]
COOL_T_OUT	Temperatura sortida motor [ºC]
CURRENT	Corrent [A]
E_ENERGY	Energia elèctrica generada [kWh]
E_POWER	Potència elèctrica generada [W]
HEAD_T_SET	Temperatura màxima de funcionament receptor [ºC]
HEAT_T_CON	Temperatura receptor [ºC]
HRAT_T_LIM	Temperatura límit de seguretat receptor [ºC]
VOLTAGE	Voltatge [V]

Taula 3: Taula de les variables obtingudes i el seu corresponent identificador [Font pròpia]

A més de les variables que s'emmagatzemaran a la base de dades, el servidor XML mostrà altres variables que no tenen tant d'interès com les anteriors, però que també seran comentades:

- **DESCRIPTION.** Mostra una descripció del dispositiu. En el nostre cas, aquest camp està buit. Aquest paràmetre seria de major interès si hi hagués més d'un dispositiu connectat a l'EDS.
- **NAME.** Indica el nom del dispositiu. En el nostre cas, TRINUM.
- **STATUS.** Indica l'estat en què es troba el dispositiu i pot prendre 5 valors diferents:
  - **1**: Comunicació correcte.
  - **4**: No inicialitzat.
  - **18**: Port incorrecte.
  - **34**: Error de comunicació.
  - **66**: Dispositiu incorrecte.
- **VDTTM.** Indica la data i l'hora de l'última comunicació amb el servidor.

A més de les variables obtingudes amb el servidor, es calcularan altres que poden ser de gran interès. Aquestes variables es mostren a la taula següent (Taula 4) juntament amb l'identificador amb el qual s'introduiran a la base de dades.

IDENTIFICADOR	VARIABLE
P_TERM	Potència tèrmica generada [W]
E_TERM	Energia tèrmica generada [kWh]
PERC_ELEC	Potència elèctrica sobre la nominal [%]
PERC_TERM	Potència tèrmica sobre la nominal [%]

Taula 4: Taula de les variables calculades i el seu corresponent identificador [Font pròpia]

Per calcular les variables mostrades anteriorment, s'ha emprat les equacions que s'exposen a continuació. Per calcular la potència tèrmica generada s'ha utilitzat:

$$P_t = \Delta \mathrm{T} \cdot C_w \cdot \mathrm{Q}$$

Per calcular aquesta equació amb les variables obtingudes, s'ha d'aplicar un canvi en les unitats del cabal del fluid refrigerant perquè les unitats siguin l/s. La fórmula per calcular la potència tèrmica generada quedaria de la següent manera:

$$\mathrm{P\_TERM} = (\mathrm{COOL\_T\_OUT} - \mathrm{COOL\_T\_IN}) \cdot C\_w \cdot \frac{COOL\_FLOW}{60}$$

On C_w és la capacitat calorífica del fluid refrigerant. En el cas del dish Stirling, el fluid refrigerant és aigua i, per tant, el valor d'aquesta variable és aproximadament 4184 [J/kg·K].

Per calcular l'energia tèrmica generada n'hi ha prou amb integrar la potència tèrmica generada en el temps. S'ha de tenir en compte que, igual que l'energia elèctrica generada, a la base de dades s'emmagatzemarà el valor acumulat d'energia que hi ha a cada instant. És a dir, a l'instant 0 (instant de la primera fila de valors que s'afegirà a la base de dades) l'energia tindrà un valor igual a 0. Després, el valor de l'energia tèrmica de la segona fila (instant 1) serà el resultat de multiplicar la potència tèrmica de l'instant 0 pel temps que ha passat entre el primer i el segon instant de la base de dades (es considera que la potència tèrmica és constant entre els dos instants).

Per calcular els percentatges de potència tèrmica i elèctrica s'ha utilitzat l'equació següent:

$$\% = \frac{P}{P_{nominal}} \cdot 100$$

La fórmula amb les variables calculades quedaria de la següent manera:

$$\mathrm{PERC\_ELEC} = \frac{E\_POWER}{P_{e\_nominal}} \cdot 100$$

$$\mathrm{PERC\_TERM} = \frac{P\_TERM}{P_{t\_nominal}} \cdot 100$$

On $P_{e\_nominal} = 1$ kW i $P_{t\_nominal} = 3$ kW. Per tant, les fórmules anteriors comparen la potència tèrmica i elèctrica generada amb la que haurien de generar idealment (que correspon al seu valor nominal).

Un cop explicada l'obtenció de les variables del dish Stirling TRINUM i els seus valors, s'exposarà com s'han emmagatzemat aquests valors en una base de dades per a la seva posterior manipulació i visualització.

5 Emmagatzematge de les variables. MariaDB

Les variables obtingudes i les variables calculades esmentades a les taules anteriors (Taules 3 i 4) s'han emmagatzemat emprant un sistema de gestió de base de dades anomenat MariaDB. Aquest sistema és una derivació del conegut MySQL que és la base de dades de codi obert més popular del món. MariaDB utilitza el mateix codi font que MySQL, però és una versió millorada i amb canvis respecte al sistema de gestió de base de dades original. El sistema MariaDB serveix per emmagatzemar tota la informació necessària en bases de dades i permet administrar aquestes dades de manera simple gràcies a la seva interfície visual i a les eines que inclou i que s'explicaran a continuació.

5.1 Llenguatge de programació. SQL

SQL (Structured Query Lenguage, en català Llenguatge de Consulta Estructurada) és un llenguatge de programació que ajuda a la resolució de problemes relacionats amb la definició i manipulació de la informació de les dades que integren una base de dades. El SQL és un llenguatge d'*alt nivell* o *de no procediment*, això implica que una única comanda SQL podria equivaldre a una gran quantitat de línies de programació en un llenguatge de programació de menys nivell. Aquest fet permet una alta producció de codificació i simplifica els treballs relacionats amb la gestió de dades. Les característiques principals d'aquest llenguatge són:

- **Llenguatge de definició de dades (LDD).** Inclou comandes per a la definició i manipulació de l'estructura o esquema de la base de dades.
- **Llenguatge interactiu de manipulació de dades (DML).** Incorpora comandes que fan possible les següents accions: inserir dades en taules, consultar les dades de les taules, actualitzar les dades contingudes en aquestes taules i eliminar dades de les taules.
- **Integritat.** Té comandes que permeten especificar les restriccions d'integritat de les bases de dades.
- **Definició de vistes.** S'inclouen comandes per definir les vistes.
- **Control de transaccions.** Es pot especificar l'inici i el final d'una transacció.
- **Autorització.** Inclou comandes per atorgar els privilegis d'accés a les bases de dades.
- **SQL incorporat i dinàmic.** Es poden incorporar comandes de SQL en altres llenguatges de programació com: C++, C, Java...

A més, l'escriptura del codi en aquest llenguatge de programació es caracteritza perquè, tradicionalment, les comandes s'escriuen en majúscules i la resta del codi, com per exemple noms de taules, en minúscula. Tot i això, el codi no distingeix entre majúscules i minúscules per la qual cosa no és estrictament necessari seguir aquest conveni, tot i que sí que és molt recomanable. El que si s'ha d'afegir sempre és un punt i coma (;) al final de cada línia de codi perquè aquest s'executi correctament. [20]

Dins de les comandes del Llenguatge de Definició de Dades destaquen 4 operacions bàsiques que, com s'explicarà més endavant a l'apartat 5.2, es faran servir per la creació i manipulació de la base de dades necessària per a l'execució d'aquest projecte. Aquestes quatre operacions són:

- **CREATE.** Permet crear taules i bases de dades.
- **ALTER.** Permet modificar l'estructura de taules realitzant accions com afegir o treure camps a la taula i modificar el tipus d'un camp.
- **DROP.** Permet eliminar objectes de les bases de dades.
- **TRUNCATE.** Permet eliminar el contingut d'una taula.

5.2 Creació i manipulació de la base de dades

En el cas d'aquest projecte s'han realitzat totes les operacions referents a la creació i modificació de la base de dades a la Consola del sistema de Windows o Indicador d'ordres (CMD). Un cop instal·lat el sistema MariaDB, aquest s'inicialitza amb la comanda: *mysql -u root -p* (on root és l'usuari) i posteriorment introduint la contrasenya escollida. Havent inicialitzat MariaDB el següent pas consisteix en la creació de la base de dades amb la comanda: CREATE DATABASE <nom de la base de dades>. Posteriorment, es poden veure totes les bases de dades existents amb la comanda: SHOW DATABASES i per utilitzar una base de dades s'introdueix la comanda: USE <nom de la base de dades>. Es pot veure tot aquest procediment a la imatge inferior (Fig. 10).

```
Microsoft Windows [Versión 10.0.19042.867]
(c) 2020 Microsoft Corporation. Todos los derechos reservados.

C:\Users\victo>mysql -u root -p
Enter password: *********
Welcome to the MariaDB monitor.  Commands end with ; or \g.
Your MariaDB connection id is 8
Server version: 10.5.9-MariaDB mariadb.org binary distribution

Copyright (c) 2000, 2018, Oracle, MariaDB Corporation Ab and others.

Type 'help;' or '\h' for help. Type '\c' to clear the current input statement.

MariaDB [(none)]> CREATE DATABASE dishstirling;
Query OK, 1 row affected (0.003 sec)

MariaDB [(none)]> SHOW DATABASES;
+--------------------+
| Database           |
+--------------------+
| dishstirling       |
| information_schema |
+--------------------+
2 rows in set (0.001 sec)

MariaDB [(none)]> USE dishstirling;
Database changed
```

Figura 10: Procediment per la creació i utilització d'una base de dades en MariaDB [Font Pròpia]

Dins de la mateixa base de dades que s'utilitzarà (anomenada "dishstirling") s'ha de crear una taula on s'introduiran totes les dades. Per crear-la s'ha d'escollir el nom de la taula i el nom de les columnes on s'emmagatzemaran cadascuna de les variables. Per fer-ho s'utilitza la comanda CREATE TABLE <nom de la taula> i posteriorment es defineixen els noms i les característiques de les columnes com es veu a la imatge inferior (Fig.11).

```
MariaDB [dishstirling]> CREATE TABLE trinum(
    -> id INT(11) NOT NULL AUTO_INCREMENT,
    -> AMB_T FLOAT(10) NOT NULL,
    -> BOILER_T FLOAT(10) NOT NULL,
    -> COOL_FLOW FLOAT(10) NOT NULL,
    -> COOL_T_IN FLOAT(10) NOT NULL,
    -> COOL_T_OUT FLOAT(10) NOT NULL,
    -> CURRENT FLOAT(10) NOT NULL,
    -> E_ENERGY FLOAT(10) NOT NULL,
    -> E_POWER FLOAT(10) NOT NULL,
    -> HEAT_T_CON FLOAT(10) NOT NULL,
    -> HEAT_T_SET FLOAT(10) NOT NULL,
    -> HRAT_T_LIM FLOAT(10) NOT NULL,
    -> VOLTAGE FLOAT(10) NOT NULL,
    -> P_TERM FLOAT(10) NOT NULL,
    -> PERC_ELEC FLOAT(10) NOT NULL,
    -> PERC_TERM FLOAT(10) NOT NULL,
    -> epoch BIGINT(13) NOT NULL,
    -> STATUS INT(10) NOT NULL,
    -> VDTTM FLOAT(20) NOT NULL,
    -> PRIMARY KEY (id)
    -> );
Query OK, 0 rows affected (0.053 sec)
```

Figura 11: Procediment per la creació d'una taula dins de la base de dades [Font Pròpia]

La taula s'ha anomenat "trinum" i s'ha afegit una columna per a cadascuna de les variables que apareixien a les taules 3 i 4, indicant com a nom de la columna el codi que apareixia a la columna "IDENTIFICADOR" de les taules. A més, s'ha considerat oportú afegir les variables VDTTM i STATUS, ja que, tot i no estar pensat que apareguin al dashboard, pot resultar interessant introduir-les per si en algun moment es vol consultar el seu valor. També s'ha afegit la columna *epoch*, on s'introduirà un valor de 13 dígits que indica l'instant de temps en el qual s'han inclòs cadascun dels valors de les variables. D'aquesta manera es podrà, posteriorment, mostrar les variables en un gràfic i visualitzar la seva evolució temporal.

Cadascuna de les files de la figura 11 indica la creació d'una columna. Al final de totes les files apareix la comanda NOT NULL que indica que la columna prendrà valors no nuls. A més, com es pot veure a la imatge, el valor de les variables seran INT (nombre enter), FLOAT (nombre real) o BIGINT (nombre enter de gran longitud). Al costat d'aquesta definició dels valors que ompliran les columnes, apareix entre parèntesis un valor que indica la longitud màxima que poden prendre les variables que s'inclouran a les columnes. Finalment, s'explicarà la primera variable ("id"). Aquesta variable és exclusivament un indicador que s'ha inclòs per definir i ordenar les files un cop introduïts els valors de les variables. A més, permet poder accedir a diferents columnes de forma molt simple i al final s'ha inclòs la comanda AUTO_INCREMENT que fa que el seu valor incrementi de manera automàtica sense haver d'afegir-lo manualment.

Per tal que s'entengui totalment l'explicació de la creació de cadascuna de les columnes de la taula, s'ha afegit la imatge inferior (Fig. 12) amb l'explicació de cada part d'una de les files que apareix a la figura 11 (la creació de la columna AMB_T).

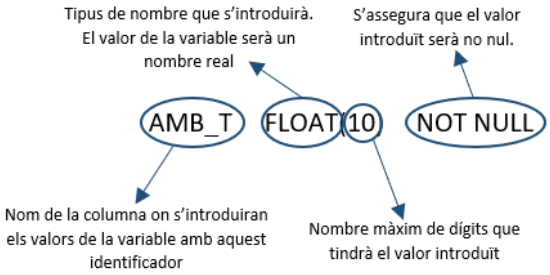


Figura 12: Explicació de les comandes per crear la columna AMB_T [Font Pròpia]

Un cop creada la taula, es pot veure un resum de les seves característiques amb la comanda DESCRIBE <nom de la taula>. Una altra manera d'obtenir la mateixa informació és amb la comanda SHOW COLUMNS FROM <nom de la taula>. També es poden veure les taules que hi ha dins de la base de dades creada amb la comanda SHOW TABLES (Fig. 13).

```
MariaDB [dishstirling]> DESCRIBE trinum;
+------------+------------+------+-----+---------+----------------+
| Field      | Type       | Null | Key | Default | Extra          |
+------------+------------+------+-----+---------+----------------+
| id         | int(11)    | NO   | PRI | NULL    | auto_increment |
| AMB_T      | float      | NO   |     | NULL    |                |
| BOILER_T   | float      | NO   |     | NULL    |                |
| COOL_FLOW  | float      | NO   |     | NULL    |                |
| COOL_T_IN  | float      | NO   |     | NULL    |                |
| COOL_T_OUT | float      | NO   |     | NULL    |                |
| CURRENT    | float      | NO   |     | NULL    |                |
| E_ENERGY   | float      | NO   |     | NULL    |                |
| E_POWER    | float      | NO   |     | NULL    |                |
| HEAT_T_CON | float      | NO   |     | NULL    |                |
| HEAT_T_SET | float      | NO   |     | NULL    |                |
| HRAT_T_LIM | float      | NO   |     | NULL    |                |
| VOLTAGE    | float      | NO   |     | NULL    |                |
| P_TERM     | float      | NO   |     | NULL    |                |
| PERC_ELEC  | float      | NO   |     | NULL    |                |
| PERC_TERM  | float      | NO   |     | NULL    |                |
| epoch      | bigint(13) | NO   |     | NULL    |                |
| STATUS     | int(10)    | NO   |     | NULL    |                |
| VDTTM      | float      | NO   |     | NULL    |                |
+------------+------------+------+-----+---------+----------------+
19 rows in set (0.064 sec)

MariaDB [dishstirling]> SHOW TABLES;
+------------------------+
| Tables_in_dishstirling |
+------------------------+
| trinum                 |
+------------------------+
1 row in set (0.001 sec)
```

Figura 13: Descripció de la taula 'trinum' i taules existents a la base de dades [Font Pròpia]

Per poder visualitzar totes les dades emmagatzemades en una taula s'ha d'usar la comanda SELECT * FROM <nom de la taula> (on el símbol * equival a *all*). Si només es vol que apareguin algunes de les columnes es pot usar la comanda de la següent manera: SELECT <nom columna 1>, <nom columna 2>, <nom columna n> FROM <nom de la taula>. S'han d'introduir els noms de totes les columnes que es vol que apareguin separats per una coma (Fig. 14).

```
MariaDB [dishstirling]> SELECT * FROM trinum;
+----+-------+----------+-----------+-----------+------------+---------+----------+---------+------------+------------+------------+---------+--------+----------+---------
--+---------------+--------+---------+
| id | AMB_T | BOILER_T | COOL_FLOW | COOL_T_IN | COOL_T_OUT | CURRENT | E_ENERGY | E_POWER | HEAT_T_CON | HEAT_T_SET | HRAT_T_LIM | VOLTAGE | P_TERM | PERC_ELEC | PERC_TER
M | epoch         | STATUS | VDTTM         |
+----+-------+----------+-----------+-----------+------------+---------+----------+---------+------------+------------+------------+---------+--------+----------+---------
--+---------------+--------+---------+
|  1 |  15.2 |     14.2 |      12.6 |      13.5 |       13.6 |       0 |   1052.8 |       0 |         16 |        525 |         16 |       0 | 87.864 |        0 |    2.928
8 | 1617812617278 |      1 | 7042020000000 |
|  2 |  14.9 |     14.2 |      12.6 |      13.3 |       13.4 |       0 |   1052.8 |       0 |         16 |        525 |         16 |       0 | 87.864 |        0 |    2.928
8 | 1617813198989 |      1 | 7042020000000 |
|  3 |  14.4 |     14.2 |      12.5 |      13.2 |       13.2 |       0 |   1052.8 |       0 |         16 |        525 |       16.1 |       0 |      0 |        0 |
0 | 1617814755827 |      1 | 7042020000000 |
|  4 |  14.2 |     14.2 |      12.6 |      13.3 |       13.4 |       0 |   1052.8 |       0 |         16 |        525 |         16 |       0 | 87.864 |        0 |    2.928
8 | 1617815124186 |      1 | 7042020000000 |
|  5 |  14.2 |     14.2 |      12.5 |      13.2 |       13.2 |       0 |   1052.8 |       0 |       16.1 |        525 |         16 |       0 |      0 |        0 |
0 | 1617815724205 |      1 | 7042020000000 |
|  6 |  13.9 |     14.2 |      12.5 |      13.2 |       13.2 |       0 |   1052.8 |       0 |       16.3 |        525 |       16.3 |       0 |      0 |        0 |
0 | 1617816324170 |      1 | 7042020000000 |
+----+-------+----------+-----------+-----------+------------+---------+----------+---------+------------+------------+------------+---------+--------+----------+---------
--+---------------+--------+---------+
6 rows in set (0.001 sec)

MariaDB [dishstirling]> SELECT AMB_T, BOILER_T FROM trinum;
+-------+----------+
| AMB_T | BOILER_T |
+-------+----------+
|  15.2 |     14.2 |
|  14.9 |     14.2 |
|  14.4 |     14.2 |
|  14.2 |     14.2 |
|  14.2 |     14.2 |
|  13.9 |     14.2 |
+-------+----------+
6 rows in set (0.001 sec)
```

Figura 14: Selecció de dades de la taula 'trinum' [Font Pròpia]

A més, es poden aplicar condicions perquè només apareguin algunes dades amb la comanda SELECT * FROM <nom de la taula> WHERE <condició lògica>. La comanda WHERE es pot combinar amb moltes altres comandes i ofereix la possibilitat de realitzar operacions només a les files que compleixin una certa condició. Es pot veure un exemple a la següent imatge (Fig. 15).

```
MariaDB [dishstirling]> SELECT id, AMB_T, BOILER_T FROM trinum WHERE COOL_T_IN < 13.3;
+----+-------+----------+
| id | AMB_T | BOILER_T |
+----+-------+----------+
|  3 |  14.4 |     14.2 |
|  5 |  14.2 |     14.2 |
|  6 |  13.9 |     14.2 |
|  8 |  13.1 |     14.2 |
|  9 |    12 |     14.2 |
+----+-------+----------+
5 rows in set (0.001 sec)
```

Figura 15: Selecció de dades de la taula 'trinum' amb alguna condició [Font Pròpia]

Es creu convenient explicar també altres operacions que es poden fer amb les comandes SQL i que s'han utilitzat durant la realització del projecte. Es considera important esmentar aquestes operacions, ja que poden ser de rellevància en la manipulació de bases de dades: [22] [21]

- **Eliminar una base de dades**. DROP DATABASE <nom de la base de dades>.
- **Canviar el nom d'una taula**. RENAME TABLE <nom de la taula> TO <nom que es vol assignar a la taula>.
- **Eliminar una taula de la base de dades**. DROP TABLE <nom de la taula>.
- **Eliminar una columna**. ALTER TABLE <nom de taula> DROP COLUMN <nom de la columna>.

- **Afegir una columna a una taula**. ALTER TABLE <nom de la taula> ADD COLUMN <nom de la columna> <tipus de les dades de la columna>. (Al tipus de les dades de la columna s'haura de afegir per exemple INT(10) en cas de que les dades fossin un nombre enter de màxim 10 dígits).

- **Canviar el nom d'una columna**. ALTER TABLE <nom de la taula> CHANGE <nom de la columna> <nom que es vol assignar a la columna> <tipus de les dades de la columna>.

- **Afegir dades a una taula**. INSERT INTO <nom de la taula> (<nom de la columna 1>, <nom de la columna 2>, <nom de la columna n>) VALUES (<valor 1>,<valor 2>,<valor n>).

A més de les comandes explicades, SQL també inclou comandes que són habituals en altres llenguatges de programació com per exemple comandes per fer operacions algebraiques: SUM, ROUND, SQRT, MAX, MIN, AVG... Finalment, cal destacar les comandes AND i OR que seran d'utilitat quan es vulgui imposar més d'una condició a la base de dades (es poden fer servir dins de la comanda WHERE per imposar diverses condicions).

Al següent apartat (Apartat 6.2), s'explicarà com amb Node-Red s'ha creat un programa per introduir els valors de les variables a la base de dades creada amb MariaDB. Per fer-ho s'utilitzaran les comandes SQL anteriorment explicades.

6 Node-Red

La major part del treball pràctic d'aquest projecte s'ha realitzat amb la plataforma Node-Red. Amb aquesta eina de programació s'ha realitzat un programa per emmagatzemar les dades de les variables del dish Stirling en una base de dades (Apartat 6.2) y un altre programa per crear un dashboard on visualitzar l'històric de les dades (Apartat 6.3). A aquests programes se'ls anomena *FLOWS* o fluxos en català.

Node-Red va ser creat per Nick O'Leary i Dave Conway-Jones del grup de Serveis de Tecnologies Emergents d'IBM (International Business Machines Corporation) l'any 2013. El seu objectiu era donar solució a la complexitat que sorgeix quan es vol integrar el hardware de l'usuari amb altres serveis. Per explicar que és Node-Red és important conèixer el terme IoT (Internet of Things, Internet de les coses en català), concepte que fa referència a la interconnexió digital d'Internet amb objectes quotidians. Concretament, és la connexió d'Internet amb objectes en comptes d'amb persones i la seva creació va suposar una millora en serveis com l'educació, la seguretat o el transport i va permetre noves oportunitats en l'accés a dades. Dins del camp de l'IoT trobem Node-Red que és una eina de programació visual utilitzada per connectar dispositius de hardware, APIs (Application Programming Interface, en català Interfície de Programació d'Aplicacions) i serveis en línia. Node-Red és un editor de flux basat en el navegador que permet afegir nodes i connectar-los entre si amb la finalitat que es comuniquin i poder formar programes sense haver d'escriure moltes línies de codi de programació. Aquesta eina de programació s'utilitza principalment en l'IIoT (Industrial Internet of Thing, en català Internet Industrial de les coses), pel fet que simplifica els processos d'informació entre productor i consumidors i és molt útil en la gestió i processament de dades en temps real.

Node-Red és un entorn de programació basat en l'entorn Node.js (Apartat 6.1.2) i permet connectar gràficament blocs predefinits, anomenats nodes, per desenvolupar una tasca concreta. La connexió entre nodes d'entrada, de sortida i de processament, formen el que s'anomena FLOW, on es formen els programes desitjats. Cadascun d'aquests blocs anomenats nodes són l'estructura mínima de Node-Red i equivalen a funcions de JavaScript (Apartat 6.1.1). A més, s'uneixen per enviar i rebre informació entre ells. A aquest mètode se l'anomena *Flow-Based Programming* i facilita molt el treball, ja que s'evita l'escriptura de moltes línies de programació amb la simple col·locació d'un node equivalent. Per tant, el funcionament de Node-Red es basa en la unió de nodes per formar programes. Entre aquests nodes es transmetrà informació a la qual se l'anomena missatge (msg) i de la qual s'ha de destacar les següents propietats:

- **msg.payload:** Representa el cos del missatge, que pot ser qualsevol mena d'objecte de JavaScript (matrius, valors, strings, etc.). És la propietat que utilitzen la majoria de nodes.

- **msg.topic:** S'utilitza per identificar la font del missatge que es transmet o per identificar diferents flows de missatges en els mateixos fluxos. Algun node utilitza aquesta propietat, com per exemple el node *mysql*.

Dins dels grans avantatges de Node-Red trobem que és Open Source (de software lliure) i que disposa d'una llibreria amb més de 2500 nodes. A més, està contínuament en creixement i es creen habitualment nous nodes que poden ser utilitzats en els fluxos propis. També existeix un fòrum (Enllaç al fòrum) a la pàgina web pròpia de Node-Red on la comunitat intentar donar solució als diferents problemes dels usuaris. En aquest fòrum també es mostren flows d'exemple en format JSON (Apartat 6.1.3) que es poden inicialitzar en el propi hardware per manipular-los

i fer proves. [23] [24] Gràcies al baix consum de recursos de Node-Red i al seu lleuger *runtime*, es pot instal·lar i executar en un ampli nombre de dispositius i entorns de desenvolupament:

- **Raspberry Pi.** Una de les opcions és instal·lar Node-Red en un dispositiu de baix cost com pot ser *Raspberry Pi*, que és una placa amb memòria RAM i processador, a més d'una sèrie de connectors per expandir les capacitats i funcions de la placa.

- **Localment.** Es pot instal·lar en un ordinador o servidor local amb sistema operatiu Windows, Linux o Mac.

- **Cloud.** Una altra opció és instal·lar els flows al núvol. Aquesta opció té l'avantatge de permetre la connexió amb altres serveis que també es gestionin des del núvol.

L'accés a Node-Red es realitza des d'una interfície web, on es creen i modifiquen els fluxos per després compilar i córrer el codi. En el cas d'aquest projecte, es treballarà en un dispositiu local. Això té el problema que Node-Red només executarà el codi quan l'ordinador estigui encès i, per tant, no s'agafaran dades del dish Stirling ni s'actualitzarà el Dashboard de manera periòdica. És per aquest motiu que un cop finalitzat el treball, es pretén instal·lar els flows de Node-Red en un servidor Linux que per permetre l'execució constant del codi creat.

A la imatge 16 es pot veure la interfície de Node-Red. La part central que apareix com una quadrícula és la part principal, on s'afegiran i connectaran els nodes que formaran part del flux. A la part superior de la quadrícula es pot escollir amb quin flux treballar en cas que es tingui més d'un flow creat. A l'esquerra de la imatge trobem tots els nodes de la llibreria que es tinguin instal·lats. Per instal·lar nous nodes s'ha de prémer a les tres línies de la part superior dreta (Menú) i després donar-li a Manage palette per finalment buscar els nodes que es vulguin instal·lar (Imatge 17). Un dels principals grups de nodes que s'han hagut d'instal·lar pel desenvolupament d'aquest treball ha sigut el *node-red-dashboard*. Dins del menú també es pot configurar els flows de Node-Red i importar i exportar flows. Al costat esquerre del Menú, apareix el botó *DEPLOY* que s'utilitza per executar els flows. Finalment, la barra lateral dreta s'utilitza principalment per visualitzar els missatges rebuts per un node Debug, com s'explicarà a continuació. Des d'aquesta secció de la interfície també és pot obrir i configurar el dashboard creat, habilitar o deshabilitar flows i configurar els nodes.

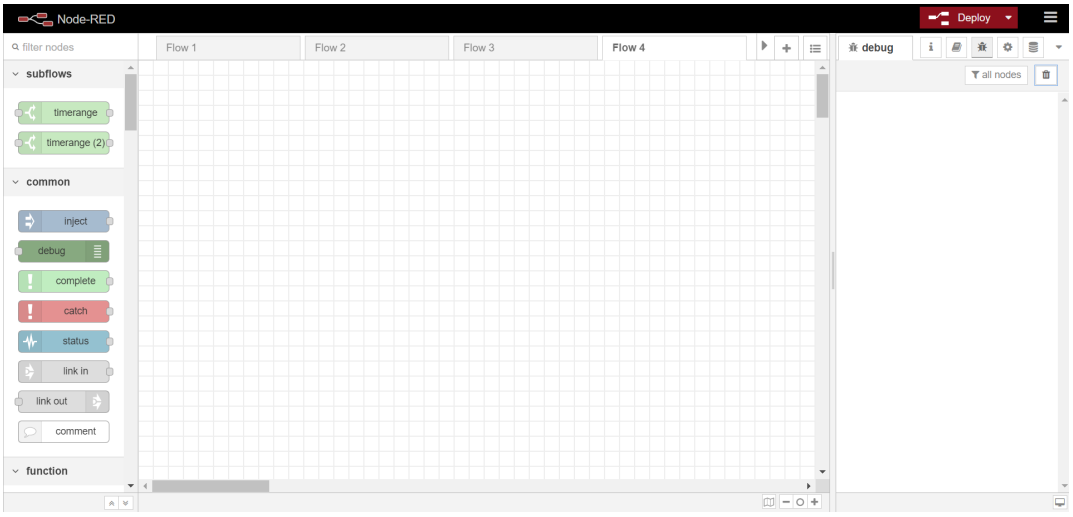


Figura 16: Interfície de programació de Node-Red [Font Pròpia]

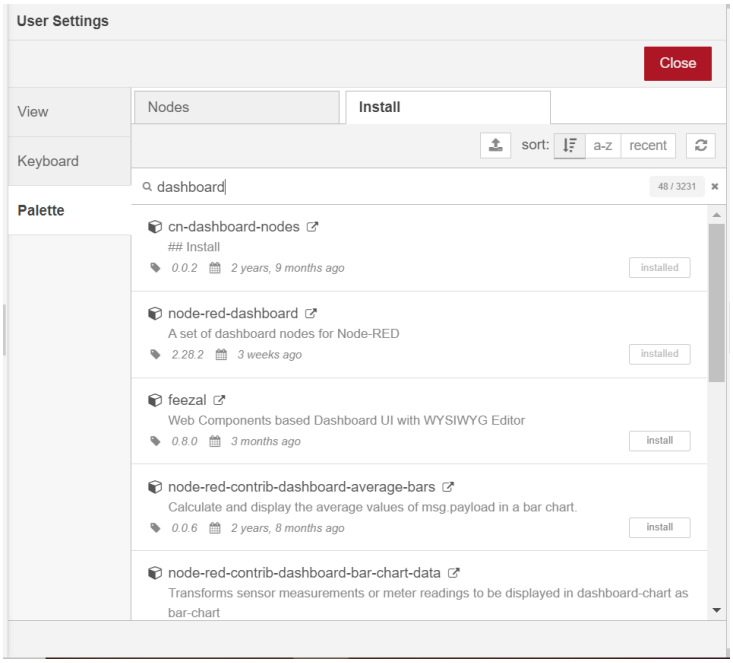


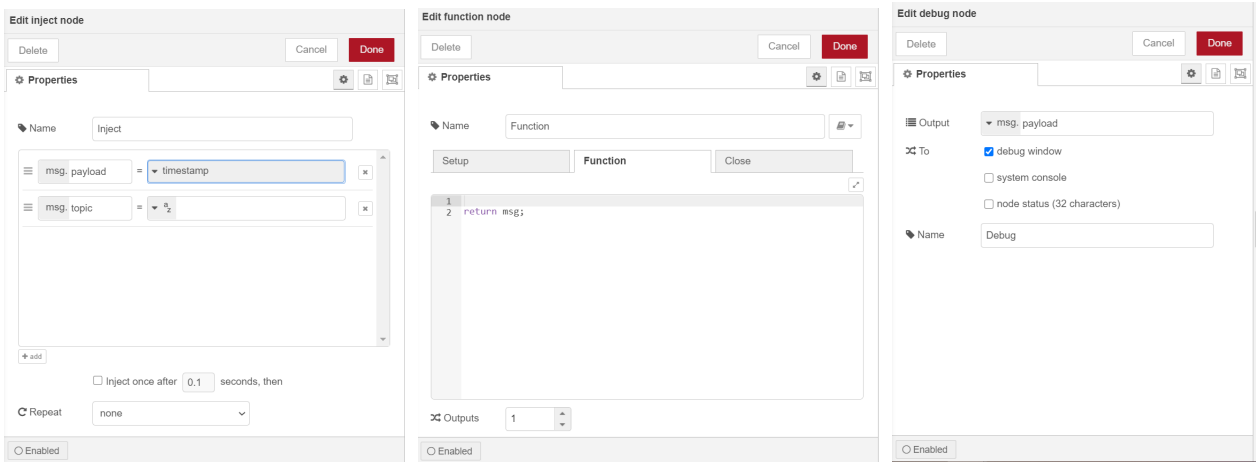
Figura 17: Exemple d'instal·lació dels nodes Dashboard de la llibreria Node-Red [Font Pròpia]

A mesura que s'expliquin els flows del present projecte, s'aniran explicant els nodes utilitzats i la seva configuració, però es creu convenient indicar prèviament el funcionament dels tres nodes principals de Node-Red (Imatge 18):

- **Inject.** El node Inject s'utilitza per activar de manera manual un flux fent clic al botó que hi ha a l'esquerra del node dins de l'editor. També es pot utilitzar per activar el flow de manera periòdica a intervals de temps regulars. Això es pot fer configurant-lo a la part inferior de la finestra del node on posa *Repeat* (Imatge 18a). A més, permet enviar qualsevol mena de missatge (payload) com pot ser un string, una expressió booleana, un missatge en format JSON, etc.

- **Function.** El node Function permet executar un codi de programació JavaScript en els missatges que es transmeten a través d'ells. El missatge passa a través d'ell com un objecte anomenat msg. Per convenció tindrà una propietat msg.payload que conté el cos del missatge. A la finestra de configuració del node Function (Imatge 18b) es pot escollir el nombre de sortides que es vol que tingui el node i és on s'introdueix el codi JavaScript.

- **Debug.** El node Debug s'utilitza per visualitzar missatges a la barra lateral dreta Debug dins de l'editor. A aquesta part de l'editor es podria veure el contingut del missatge, el tipus d'aquest, el moment en què s'ha rebut el missatge al node Debug i quin node l'ha enviat. Amb el botó del node es pot habilitar o deshabilitar la seva sortida. Aquest node és de molta utilitat, principalment, quan hi ha errors al Flow amb el que s'està treballant. La millor manera de solucionar un error és col·locar nodes Debug a parts intermèdies del flux per veure les característiques del missatge en aquests punts i comprovar si concorda amb el que s'esperava. D'aquesta manera es pot saber en quin punt del flux es troba l'error. A la finestra de configuració del node Debug (Imatge 18c) es pot escollir quina sortida es vol que aparegui a la barra lateral dreta de l'editor.



(a) Node Inject (b) Node Function (c) Node Debug

Figura 18: Configuració dels nodes principals de Node-Red [Font pròpia]

A la imatge 19 es mostra un flow d'exemple dins de l'editor de Node-Red. Es pot veure com es realitza la connexió entre nodes. Utilitzant simplement aquests tres nodes es poden crear una gran varietat de programes gràcies a la versatilitat que aporta el node Function.

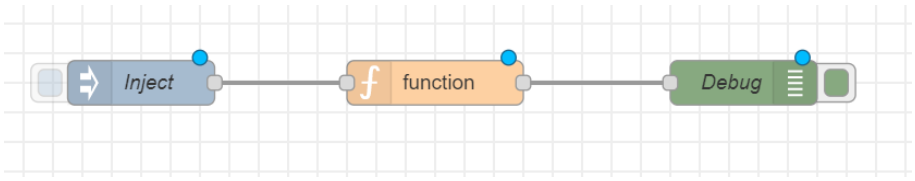


Figura 19: Exemple de flow utilitzant els nodes principals de Node-Red [Font Pròpia]

6.1 Llenguatge de programació i tecnologies necessàries

Per poder utilitzar Node-Red de manera satisfactòria i aconseguir programar correctament és imprescindible adquirir uns coneixements previs d'alguns àmbits que s'han anat comentant anteriorment. Principalment, és necessari saber programar amb el llenguatge de programació JavaScript i conèixer Node.js i JSON. A continuació, s'explicaran detalladament aquestes tres tecnologies necessàries per treballar amb Node-Red.

6.1.1 JavaScript

JavaScript (habitualment abreviat com JS) és un llenguatge de programació que permet implementar funcions complexes en pàgines web. Per tant, habitualment s'utilitza juntament amb HTML i CSS en el desenvolupament web *FrontEnd* (part d'una aplicació o web que interactua amb l'usuari). La inclusió del llenguatge JavaScript amb els altres dos llenguatges de programació permet crear elements interactius i dinàmics, millorant així la interacció dels usuaris amb la web.

JavaScript va ser desenvolupat per Brendan Eich de *Netscape Navigator* amb el nom de *Mocha* i després anomenant-se LiveScript. En el moment de la seva creació, es tenia l'objectiu d'afegir tasques del servidor en el navegador i la motivació de poder validar informació enviada al ser-

vidor, tasca que abans no es podia fer amb un bon rendiment. Més endavant, *Sun Microsystems* va comprar Netscape i li va canviar el nom al definitiu JavaScript, un nom molt semblant al llenguatge de programació Java. En aquell moment, la semblança en els noms d'ambdós llenguatges va generar molta confusió, ja que feia la impressió que JavaScript era una extensió del ja existent Java. Tot i això, és important entendre que la relació entre Java i JavaScript és únicament comercial i no tenen res a veure pel que fa a la programació.

Aquest llenguatge té la possibilitat d'oferir una àmplia funcionalitat que permet fer coses com emmagatzemar valors útils dins de variables o fer operacions sobre strings (fragments de text). Tot i això, la funcionalitat principal és la creació d'APIs (Interfícies de Programació d'Aplicacions del Navegador) construïdes en navegadors que ofereixen funcionalitats pràcticament il·limitades. Per exemple, amb la creació d'APIs es pot arribar a crear dinàmicament contingut HTML, capturar i manipular vídeos amb la càmera web de l'usuari, generar gràfics 3D, etc. [25]

Després d'haver detallat les funcionalitats i l'origen de JavaScript, aquest es pot definir com un llenguatge de programació amb les següents característiques:

- **Llenguatge del costat del client.** JavaScript s'executa al hardware del mateig usuari a través d'un navegador a diferència dels llenguatges del costat del servidor, que s'executen i interpreten pel mateix servidor i s'han de tractar abans de ser mostrats a l'usuari final.
- **Llenguatge orientat a objectes.** Utilitza classes i objectes com estructures per organitzarse de manera senzilla.
- **De tipatge dèbil o no tipatge.** Quan es declara una variable a JavaScript no fa falta especificar el tipus de dada que és. Això permet que la programació sigui més ràpida i simple.
- **D'alt nivell.** La sintaxi del llenguatge JavaScript és similar al llenguatge humà i, per tant, fàcil de comprendre. La seva sintaxi, llavors, es troba lluny del codi que utilitza una computadora per executar el que s'ha programat.
- **Llenguatge interpretat.** Com s'ha explicat a la característica anterior, el llenguatge té una sintaxi molt diferent a la sintaxi del nivell màquina. És per aquest motiu que es necessita un intèrpret que faci possible la conversió de les línies de codi al llenguatge de la màquina.

Durant la realització del present treball serà imprescindible conèixer el llenguatge de programació JavaScript, ja que com s'ha explicat anteriorment, els nodes Function de Node-Red s'han de configurar afegint un codi JavaScript que serà executat per complir amb la tasca desitjada. Per aprendre JavaScript s'ha utilitzat el llibre: "Introducción a JavaScript" [26] en la seva versió PDF disponible a internet i el llibre: "Domine JavaScript. 4a edición" en format físic [27], a més de consultar informació en diverses webs d'Internet.

6.1.2 Node.js

Node.js és un entorn de temps d'execució de JavaScript (per aquest motiu conté la terminació .js). Aquest entorn de codi obert s'utilitza per executar en temps real un programa escrit en JavaScript. Node.js va ser creat per Ryan Dahl l'any 2009 amb l'objectiu que fos útil en la creació de programes de xarxa com servidors web. A més, està orientat a esdeveniments asíncrons i dissenyat per crear aplicacions network escalables (que creix proporcionalment el rendiment de l'aplicació amb l'increment dels processadors on s'executa).

Prèviament a la creació de Node.js, gran part dels servidors webs existents tenien problemes d'escalabilitat, ja que quan es feia una petició al servidor es bloquejava l'entrada de noves peticions fins que la primera s'hagués resolt. Això dificultava molt la realització de serveis amb un gran nombre de peticions concurrents. La solució a aquest problema va ser una de les motivacions que va propulsar la creació de Node.js. L'entorn de programació va incloure un *event loop* que permet que Node.js utilitzi un model d'entrades (sol·licituds) i sortides (respostes) sense bloquejos controlat per esdeveniments, fet que fa que sigui lleuger i eficient.

Aquestes característiques de Node.js fa que sigui perfecte per la creació d'aplicacions de red ràpides, ja que pot gestionar una gran quantitat de connexions simultànies amb un gran rendiment (té una alta escalabilitat). Per contra, no és una bona opció utilitzar Node.js en operacions intensives d'un processador, ja que utilitzar-lo en programacions molt pesades impediria aprofitar els seus avantatges.

Tradicionalment, als serveis web es generava un nou subprocés per cada connexió que crea una sol·licitud ocupant la RAM del sistema. En canvi, Node.js només opera amb un subprocés, fet que millora la quantitat de RAM disponible, i aquest processa un esdeveniment rere l'altre. [28]

Node-Red basa el seu lleuger temps d'execució en Node.js, aprofitant el seu model sense bloquejos impulsat per esdeveniments. Això fa que sigui ideal per executar-se en hardwares de baix cost com Raspberry Pi o en el núvol.

6.1.3 JSON

JSON (JavScript Object Notation, en català Notació d'Objectes JavaScript) és un format de text dedicat a guardar i intercanviar informació i dades. El gran avantatge d'aquest format respecte d'altres com l'XML és que resulta molt senzill d'escriure i llegir pels programadors i molt simple d'interpretar i generar per les màquines. JSON és una alternativa amb una estructura més simple i lleugera que XML, un dels formats més utilitzats.

A principis de la dècada dels 90 existia el problema que les màquines no podien entendre's entre si, perquè habitualment utilitzaven sistemes operatius i llenguatges de programació diferents. Per solucionar-lo es va crear el format XML. Tot i solucionar els problemes existents, XML tenia problemes a l'hora de treballar amb una gran quantitat de dades, ja que el processament es tornava molt lent. Per això, a principis dels 2000 es va crear JSON com un format més lleuger i ràpid en l'intercanvi d'informació. Des d'aquest moment, JSON va permetre reduir la mida dels arxius i la quantitat de dades que s'han de transmetre. [29]

Un dels principals avantatges de JSON és que s'ha creat com un format independent de qualsevol llenguatge de programació. Això permet que els serveis que transmeten informació amb aquest format no necessitin utilitzar el mateix llenguatge de programació. És a dir, l'emissor pot utilitzar un llenguatge (per exemple Python) i el receptor un altre (per exemple C++), ja que cada llenguatge de programació té una llibreria pròpia que permet codificar i descodificar les cadenes en JSON. [30]

Per escriure correctament la sintaxis s'ha de tenir en compte que hi ha dos elements principals en un objecte JSON:

- **Keys (Claus).** És una cadena de caràcters (string). Contenen una seqüència de caràcters entre cometes dobles (no serveixen cometes simples).

- **Values (Valors).** Són un tipus de dades JSON vàlides. Poden tenir moltes formes que s'explicaran a continuació.

A més, és important saber que els objectes JSON es redacten entre dues claus {}. Dins d'aquestes claus es poden afegir diversos parells de Key/Value separats per una coma. Les claus se segueixen amb dos punts per separar-les dels valors. Un exemple de sintaxi d'un objecte JSON podria ser:

{"Temperatura": "24", "Estat": "Connectat"}

Com s'ha comentat anteriorment, el valor d'un objecte JSON pot contenir diferents tipus de dades JSON vàlides com: [31]

- **Array.** Un array és una col·lecció ordenada de valors. Es pot traduir al català com matriu o vector. Per escriure un objecte que sigui un array s'haurà d'escriure entre claudàtors [] i els valors que inclogui l'array hauran d'estar separats per comes.

- **Objecte.** Un valor també pot ser un objecte JSON dins del qual hi haurà una clau i un altre valor. En cas que el valor sigui un objecte s'haurà d'escriure entre claus {}.

- **String.** És una cadena de caràcters. S'ha d'escriure entre cometes dobles.

- **Nombre.** Pot ser un nombre enter o un float per representar nombres decimals. Tot i ser un nombre, s'ha d'escriure entre cometes com s'ha vist a l'exemple de sintaxi anterior.

- **Booleà.** Es poden utilitzar els valors "True" (vertader) o "False" (Fals) com a valor d'un objecte.

- **Nul.** S'utilitza per indicar que la clau no conté informació. Es defineix com: "null".

És important conèixer el format JSON perquè és el format utilitzat per Node-Red per transmetre els missatges entre nodes. Fins i tot, existeix un node anomenat "JSON" que permet transformar un missatge del format JSON a l'objecte JavaScript que representa i viceversa. A més, Node-Red utilitza el format JSON per importar i exportar flows, fet que facilita poder provar flows trobats per Internet que poden ser una bona ajuda per aconseguir els flows desitjats.

6.2 Flow 1. Introducció dels valors de les variables a la base de dades

Després d'explicar els coneixements previs que es van haver d'adquirir per utilitzar Node-Red, es comentarà el treball que s'ha realitzat amb aquest software. El treball s'ha dividit en dues parts que es corresponen a dos flows de Node-Red. El primer flow, a partir d'ara flow 1, correspon a la introducció dels valors de les variables a la base de dades creada amb MariaDB i el segon flow, a partir d'ara flow 2, correspon a la creació i personalització del dashboard on es visualitzaran les dades obtingudes.

Per explicar el treball fet amb Node-Red amb cadascun dels flows s'afegirà una imatge de la interfície de programació de Node-Red amb el flow corresponent i s'explicaran els nodes que apareixen en ell.

També, abans de començar l'explicació dels dos flow, es farà una descripció d'alguns nodes que sortiran freqüentment als flows i que no es van explicar a l'apartat 6 a la part de nodes principals.

Aquests nodes es poden veure a la imatge 20a i són els següents:

- **Change.** El node Change pot ser utilitzat per modificar les propietats del missatge que li arriba. Concretament té quatre funcions:
  - **Set.** Permet configurar una propietat. El valor pot ser de diferents tipus i es pot extreure d'un missatge existent.
  - **Change.** Permet buscar i canviar propietats d'un missatge.
  - **Move.** Permet moure o canviar el nom d'una propietat.
  - **Delete.** Permet eliminar una propietat.

  Aquest node s'utilitzarà principalment per emmagatzemar propietats de missatges i poder utilitzar-los més endavant. Tal com es pot veure a la figura 20b a la part esquerra (on posa *set*) es pot escollir la funció que es vol que faci el node i a la part central es pot escollir on es vol emmagatzemar una propietat.

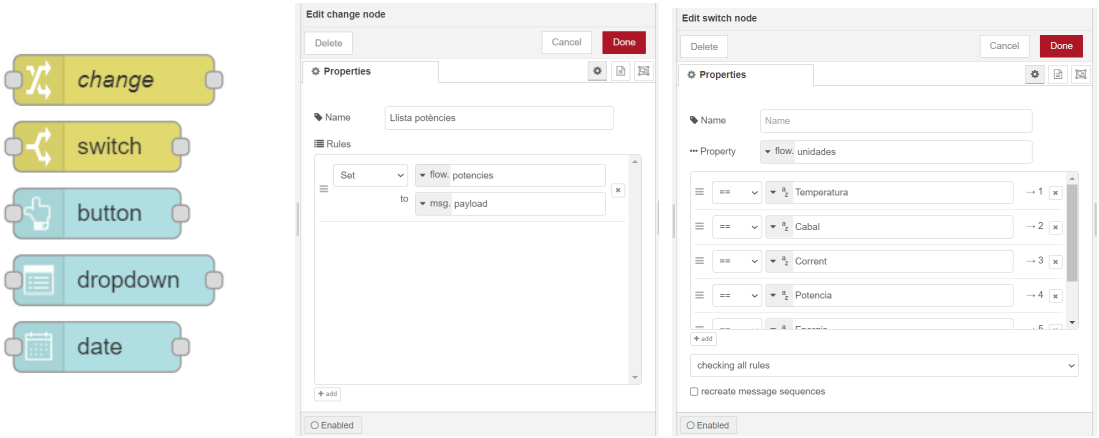
- **Switch.** Aquest node permet escollir la direcció que ha de prendre la transmissió d'un missatge a diferents branques d'un flux en funció d'alguna condició. Com s'observa a la figura 20c es defineix un llistat de condicions (en el cas de la imatge inferior el text inclòs a la propietat ha de ser igual al text que s'indica a cada condició) que s'han d'aplicar sobre una propietat definida (en el cas de la imatge inferior la propietat és el flow "unidades").

- **Button.** És un node de la llibreria dashboard que permet afegir un botó al *dashboard* elaborat. Com es veu a la part inferior de la imatge 21a, es pot configurar el msg.payload y el msg.topic que es vol que enviï el node quan aquest és pulsat al dashboard.

- **Dropdown.** És un node de la llibreria *dashboard* que permet afegir una llista d'elements que es poden seleccionar al dashboard. A la figura 21b, es pot veure la configuració d'aquest node. A la part central es defineixen tots els elements que han de sortir al desplegable del dashboard. El text de la dreta és el que es mostrarà al dashboard i el text de l'esquerra és el msg.topic que enviarà el node quan se seleccioni algun dels elements de la llista. A més, es pot decidir entre permetre una selecció múltiple o si només es pot escollir un dels elements de la llista al dashboard.

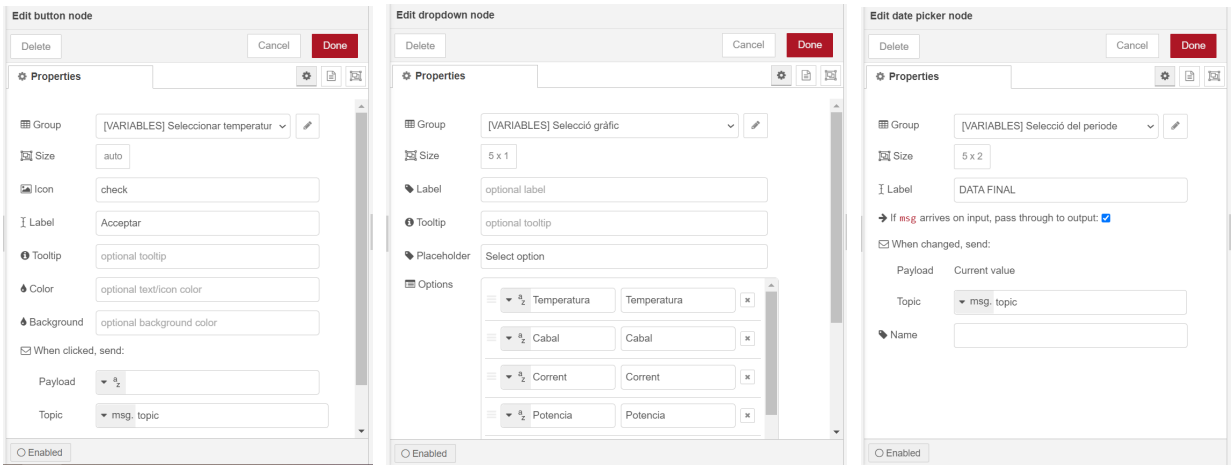
- **Date picker.** És un node de la llibreria *dashboard* que permet escollir una data en un calendari. El node envia en el msg.payload l'*epoch* de la data escollida al calendari del dashboard.

A la part superior de tots els nodes de la llibreria *dashboard* s'ha d'escollir a quina pantalla i a quin grup (els elements s'han d'agrupar a cada pantalla formant grups) del dashboard pertanyeran. També permet definir la mida del que es mostrarà al dashboard. Això facilitarà més endavant l'organització i la disposició dels elements al dashboard.



(a) Nodes utilitzats (b) Node change (c) Node switch

Figura 20: Nodes utilitzats (1) [Font pròpia]



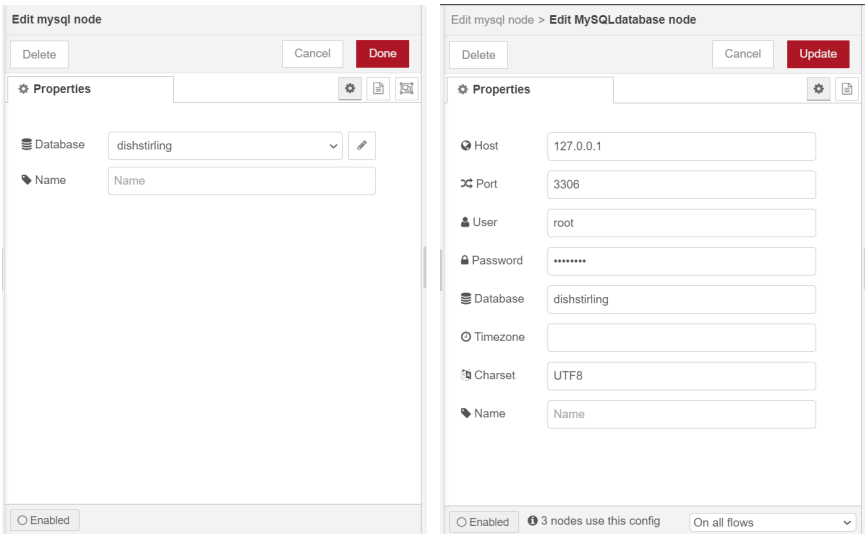
(a) Node button (b) Node dropdown (c) Node date picker

Figura 21: Nodes utilitzats (2) [Font pròpia]

Finalment, s'explica el node mysql, que apareixerà a tots els flows fets a Node-Red:

- **Node mysql:** És un node que permet l'accés bàsic a una base de dades MySQL i, per tant, llegir i escriure en la base de dades. Cal destacar que al node *mysql* ha d'arribar la consulta sostinguda al msg.topic i retornarà el resultat al msg.payload. Normalment, el resultat que surt del node és una matriu amb les dades extretes de la base de dades. La configuració d'aquest node es basa únicament a introduir la base de dades. Com es veu a la figura 22a a la primera pantalla es pot escollir el nom del node i habilitar-lo o deshabilitar-lo com a la resta de nodes. El camp principal que s'ha d'omplir és l'apartat ´´Database´´. Per fer-ho, s'ha de seleccionar una base de dades pressionant a la fletxa de la dreta, que obrirà un desplegable amb totes les bases de dades que hi hagi registrades. En cas que no s'hagi introduït la base de dades mai, s'ha de pressionar la icona del llapis de la dreta i s'obre la pantalla de configuració que es veu a la figura 22b. En aquesta pantalla s'han d'omplir totes les dades relacionades amb la configuració del servidor on està la

base de dades (Host, Port, User, Password), en el nostre cas MariaDB. Finalment, s'indica el nom de la base de dades a l'apartat "Database" i al camp "charset" s'afegeix el format de codificació que utilitza la base de dades per guardar internament les dades (en aquest cas UTF-8).



(a) Pantalla 1 del node mysql (b) Pantalla 2 del node mysql

Figura 22: Configuració del node mysql [Font pròpia]

El primer flow consisteix en la introducció dels valors de les variables del dish Stirling en la base de dades que es va crear amb MariaDB (apartat 5.2). Per fer-ho s'han introduït les variables que aporta el dish Stirling i s'han creat noves variables, indicades a la taula 4, per també introduir-les a la base de dades.

A continuació, es mostrarà una imatge de la interfície de programació de Node-Red del flow 1 (Fig. 23). Com es pot veure a la imatge, el flow es divideix en dues parts. La part superior té la funció d'emmagatzemar els valors a la base de dades. La part inferior en canvi, com s'explicarà després, s'ha creat únicament per facilitar el càlcul de l'energia tèrmica. Seguidament, s'indicarà la configuració dels nodes que conformen aquest flow per tal de poder entendre correctament el funcionament d'aquest.

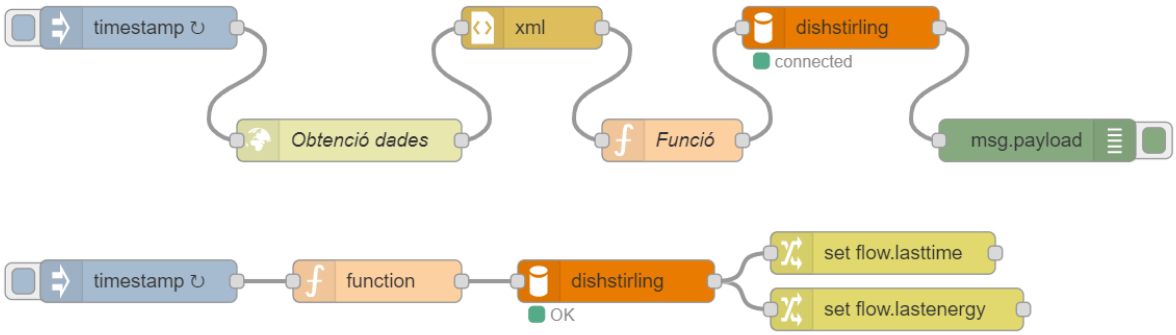


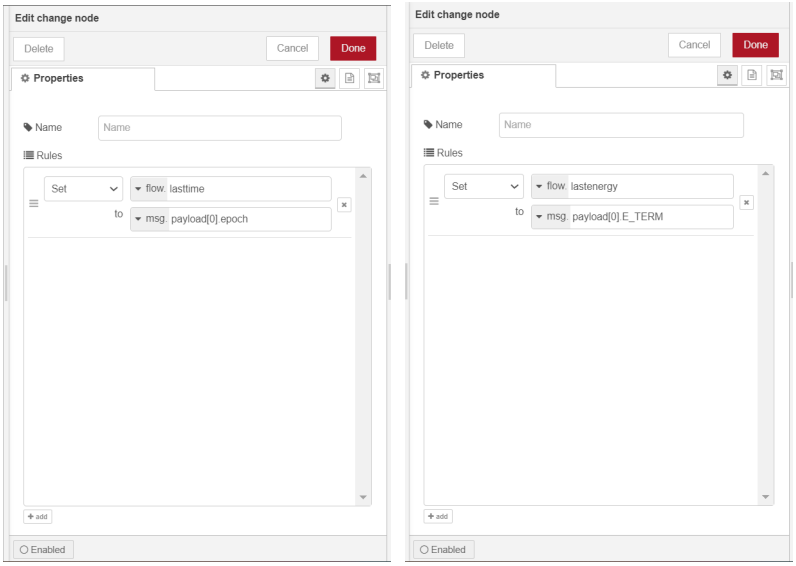
Figura 23: Flow 1 a Node-Red [Font Pròpia]

Primerament, s'explicarà de manera resumida la part inferior del flow, l'objectiu de la qual és emmagatzemar l'últim valor de l'energia tèrmica i l'epoch en què es té aquesta energia. És important emmagatzemar aquests valors, ja que, com s'ha explicat a l'apartat 4.1, per calcular el valor de l'energia tèrmica generada a cada instant fa falta tenir el valor de l'instant anterior. A la part inferior, el missatge és generat per un node *inject* i arriba a un node *function*. Més endavant, arriba a un node *mysql* per finalment dividir-se el missatge en dues parts per arribar a dos nodes *change*.

El node inject s'ha configurat per què inicïi el missatge cada 1 segon per tal de poder tenir l'últim valor de l'energia tèrmica actualitzat constantment. Pel que fa al node *function* inclou el codi següent:

$$msg.topic = "SELECT\ epoch,\ E\_TERM\ FROM\ trinum\ ORDER\ BY\ id\ DESC\ LIMIT\ 1"$$

Amb aquesta línia de codi se selecciona l'epoch i l'energia tèrmica generada de la base de dades trinum (SELECT epoch, E_TERM FROM trinum) i s'agafa només l'últim valor (ORDER BY id DESC LIMIT 1). Com s'ha mencionat abans, la línia de codi es defineix al msg.topic en comptes d'en el msg.payload perquè és d'aquesta manera com el node *mysql* (el node següent) detecta les sentències que li arriben. Després d'arribar el missatge al node *mysql*, d'aquest surt el valor de l'epoch i de l'E_TERM (la variable "energia tèrmica generada"). Aquests valors s'emmagatzemaran a les propietats *flow.lasttime* i *flow.lastenergy* respectivament, tal com es veu a la figura 24. A més, a la imatge es pot veure com per accedir als valors s'ha d'indicar la sentència: *msg.payload*[*0*].<*variable*>.



(a) Propietat flow.lasttime (b) Propietat flow.lastenergy

Figura 24: Configuració dels nodes change [Font pròpia]

Continuant amb el flow 1 (Fig. 6.2), a la part superior, el missatge és generat per un node *inject* i arriba a un node *http request*. Més endavant, arriba a un node de tipus *parser* anomenat *xml* d'on surt per arribar al node *function*. Després s'envia el missatge a un node *mysql* per finalment arribar a un node *debug*. A continuació, s'explicarà la configuració utilitzada en aquests nodes i els funcionaments dels nodes que no s'han explicat a l'apartat 6.

· **Node Inject:** El missatge s'inicia amb un node *Inject*, el qual ha sigut configurat perquè el msg.payload sigui un "timestamp" (una marca de temps) i no s'ha introduït cap valor al msg.topic, ja que l'única intenció d'afegir el node *Inject* és que inicialitzi el missatge sense donar-li un valor inicial. Com es veu a la part inferior de la figura 25 s'ha configurat el missatge amb una repetició per intervals amb la finalitat que el node enviï el missatge inicialitzat al següent node (el node *http request*) cada 10 minuts. Per tant, s'ha determinat que s'introduiran cada 10 minuts els valors de les variables del dish Stirling a la base de dades. S'ha escollit aquest interval de temps, ja que es considera que les variables emmagatzemades (temperatures i energies) en condicions normals no haurien de patir grans canvis en períodes de temps inferiors als 10 minuts. Tot i això, si es considerés necessari es podria modificar l'interval de temps a la configuració del node *Inject*.

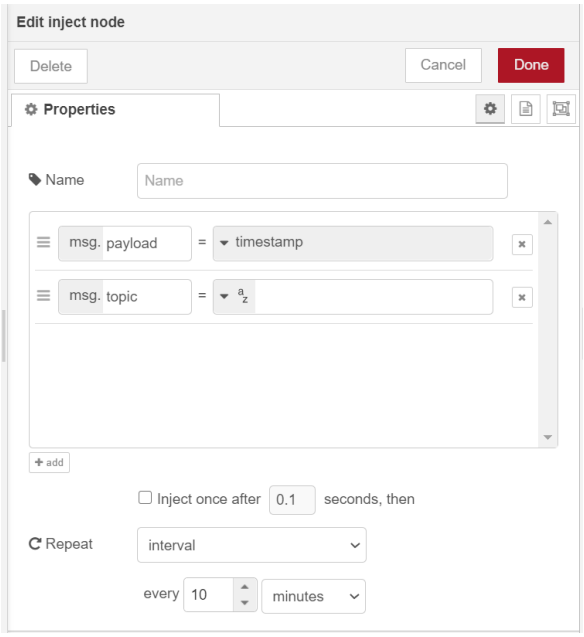


Figura 25: Configuració del node Inject [Font pròpia]

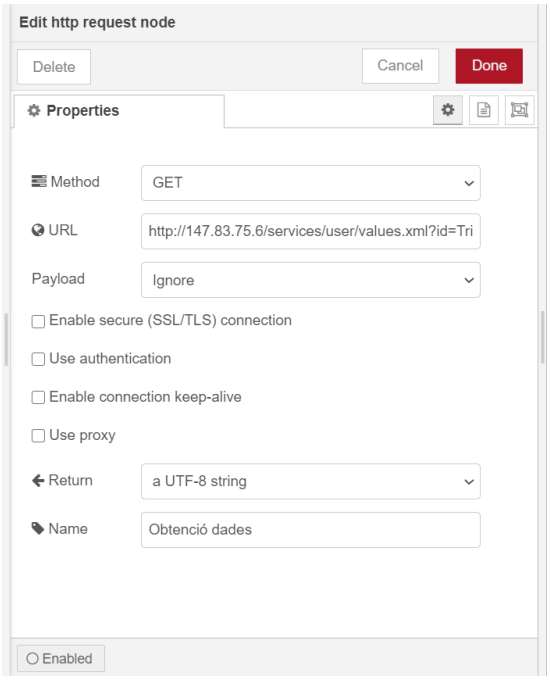


Figura 26: Configuració del node http request [Font pròpia]

· **Node http request:** Abans d'explicar la configuració d'aquest node s'explicarà la seva funcionalitat, ja que no és un dels tres nodes principals que es van explicar anteriorment a l'apartat 6. El node *http request* està inclòs al grup de nodes network. Com el seu nom indica, la seva funció consisteix a obtenir dades d'un enllaç que s'hagi introduït al node i transferir aquestes dades en forma de missatge al següent node.

Com es pot veure a la figura 26, el node *http request* té algunes característiques comunes a tots els nodes com és atorgar-li un nom al node (apartat "Name" a la part inferior de la imatge) o habilitar-lo/deshabilitar-lo (botó "Enabled" de la part inferior esquerra de la imatge). A més, a l'apartat "Method" es pot escollir la funció del node entre: GET (l'utilitzat en aquest cas per obtenir els valors), POST, PUT, DELETE i HEAD. La part principal de la configuració del node és l'apartat "URL" on s'introdueix l'enllaç que es va indicar a l'apartat 4 per obtenir els valors en XML.

· **Node XML:** El node *XML* es troba dins del grup de nodes *parser*. Aquests nodes s'utilitzen per transformar missatges en un format determinat a format JSON i a la inversa. Dins d'aquest grup trobem els nodes: csv, html, json, xml i yaml, cadascun dels quals transformen els missatges del format que indica el nom del node al format JSON. Com la resta de nodes i com es veu a la figura 27, també inclou l'apartat Name per escollir el nom del node i el botó Enabled per habilitar-lo o deshabilitar-lo.

Per tant, el node *XML* serà utilitzat per transformar el contingut de l'enllaç on s'obté els valors de les variables del dish Stirling. Aquestes dades, que es troben en format XML, es transformaran al format JSON perquè el missatge pugui ser llegit i modificat pel següent node, el node Function.

És per aquest motiu que s'ha explicat amb anterioritat, a l'apartat 6.1.3, que és el format JSON i perquè és important conèixer-lo a l'hora de fer ús de Node-Red.

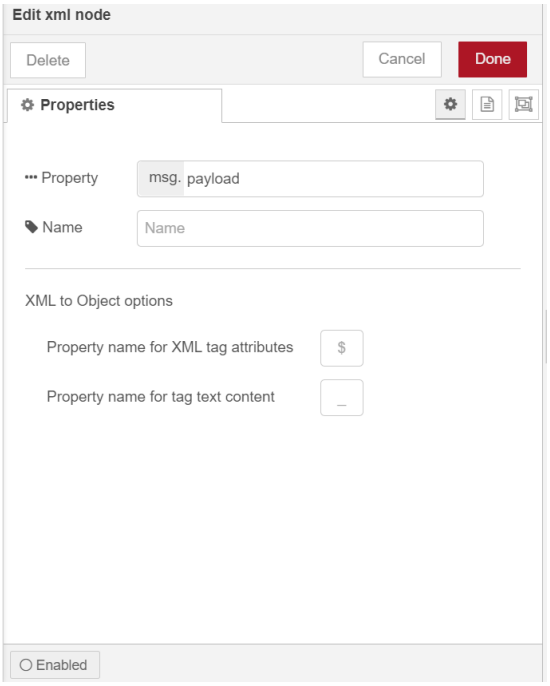


Figura 27: Configuració del node XML [Font pròpia]

```
var AMB_T = msg.payload.values.variable[0].value[0];
var BOILER_T = msg.payload.values.variable[1].value[0];
var COOL_FLOW = msg.payload.values.variable[2].value[0];
var COOL_T_IN = msg.payload.values.variable[3].value[0];
var COOL_T_OUT = msg.payload.values.variable[4].value[0];
var CURRENT = msg.payload.values.variable[5].value[0];
var E_ENERGY = msg.payload.values.variable[7].value[0];
var E_POWER = msg.payload.values.variable[8].value[0];
var HEAT_T_CON = msg.payload.values.variable[10].value[0];
var HEAT_T_SET = msg.payload.values.variable[9].value[0];
var HRAT_T_LIM = msg.payload.values.variable[11].value[0];
var VOLTAGE = msg.payload.values.variable[15].value[0];
var P_TERM = (COOL_T_OUT-COOL_T_IN)*4184*COOL_FLOW/60;

if (P_TERM < 0){
    P_TERM = 0;
}

var PERC_ELEC = E_POWER*100/1000;
var PERC_TERM = P_TERM*100/3000;
var d = new Date();
var epoch = d.getTime();
var STATUS = msg.payload.values.variable[13].value[0];
var VDTTM = msg.payload.values.variable[14].value[0];
var lastenergy = flow.get("lastenergy");
var lasttime = flow.get("lasttime");
if (lastenergy == ""){
    lastenergy = 0
}
var E_TERM = Math.round((lastenergy + (P_TERM*(epoch-lasttime)/(1000*60*60))/1000)*10000)/10000;
msg.topic = "INSERT INTO trinum(AMB_T, BOILER_T,COOL_FLOW,COOL_T_IN,COOL_T_OUT,CURRENT,E_ENERGY,E_POWER,HEAT_T_CON,HEAT_T_SET,HRAT_T_LIM,VOLTAGE,P_TERM,PERC_ELEC,\
PERC_TERM,epoch,STATUS,VDTTM,E_TERM) VALUES("+AMB_T+","+BOILER_T+","+COOL_FLOW+","+COOL_T_IN+","+COOL_T_OUT+","+CURRENT+","+E_ENERGY+","+E_POWER+","+HEAT_T_CON+",\
"+HEAT_T_SET+","+HRAT_T_LIM+","+VOLTAGE+","+P_TERM+","+PERC_ELEC+","+PERC_TERM+","+epoch+","+STATUS+","+VDTTM+","+E_TERM+")"
return msg;
```

Figura 28: Configuració del node Function [Font pròpia]

· **Node Function:** El funcionament del node *Function* ja va ser detallat a l'apartat 6, per tant, a continuació l'explicació se centrarà en el codi de programació JavaScript inclòs al node. Les primeres 30 línies del codi que es mostra a la figura 28 estan dedicades a definir variables. Cadascuna d'aquestes variables es corresponen a les variables obtingudes del dish Stirling que s'hauran d'introduir a les columnes de la base de dades amb el mateix nom. Per obtenir cadascuna de les variables del missatge en format JSON que arriba al node Function, s'ha utilitzat la següent notació:

msg.payload.values.variable[<posició de la variable>].value[0]

A la figura 29 es mostra el missatge que arriba al node function. Per visualitzar aquest missatge s'ha col·locat un node *Debug* connectat a la sortida del node *XML* (el node anterior al node *Function*). Observant aquesta imatge, resulta més senzill entendre la notació de la línia de codi anterior. La notació comença indicant d'on es vol extreure les dades, en el nostre cas del msg.payload. Després, s'accedeix a "values" per més endavant indicar la variable amb la seva posició. Dins de cada variable, com es veu a la imatge de la dreta, es divideix la informació en ´´id´´ (on s'indica el nom de la variable) i ´´value´´ (on s'indica el valor de la variable). En el nostre cas, ens interessa extreure el valor de la variable, per tant afegim ".value[0]" al final de la comanda de JavaScript.

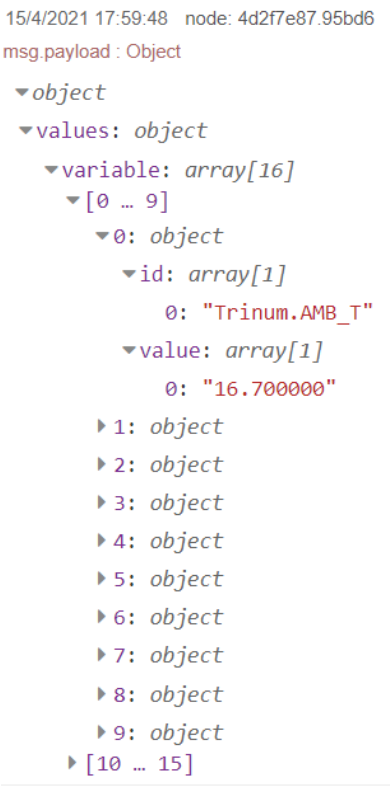


Figura 29: Variables en format JSON [Font pròpia]

Seguint amb la imatge 28, entre les definicions de les variables del codi cal destacar algunes línies. En algunes ocasions, a causa d'errors en els sensors encarregats de detectar les temperatures la potència tèrmica generada és negativa (fet que no té sentit). Per corregir això, a la línia 15 s'ha establert la condició que si la P_TERM és negativa, a la base de dades s'emmagatzemi com si fos igual a 0 W. A la línia 21 es defineix la variable *d* amb la funció *new Date*() que indica la data del moment en què s'executa el codi. Pel que fa a la línia 22, es defineix la variable epoch com l'instant de temps obtingut de la variable *d* amb la funció *d.getTime*(). A la línia 30 es defineix la variable E_TERM, la qual s'arrodoneix a 4 decimals fent: "Math.round(<valor de E_TERM>*10000)/10000".

Finalment, a la línia 31 es dóna la instrucció d'introduir els valors a la base de dades. Per fer-ho s'utilitza la comanda explicada en el punt ´´afegir dades a una taula´´ (dins de l'apartat 5.2): *INSERT INTO <nom de la taula>* (*<nom de les columnes>*) *VALUES* (*<valors>*). En aquest cas, la informació s'emmagatzema al msg.topic en comptes de fer-ho al msg.payload perquè el següent node (node *mysql*), que és el node al qual li arribarà el missatge, treballa amb el msg.topic.

Al final de la imatge del flow 1 (Figura 23) apareix un node *debug*. Aquest node no té cap rellevància en l'emmagatzematge de les variables a la base de dades i, fins i tot, es podia treure sense causar cap canvi. La seva presència, però, és convenient per poder visualitzar l'estat del missatge final i veure si s'ha realitzat tot el procés correctament. Un cop s'hagi comprovat, gràcies al node *debug* i a comprovar l'estat de la base de dades, que el funcionament és correcte, es pot eliminar aquest node sense cap problema.

6.3 Flow 2. Creació i personalització del Dashboard

El segon flow de Node-Red consisteix en l'elaboració del dashboard on es podran visualitzar i manipular les variables del dish Stirling. Tal com es veurà a continuació, el flow s'ha dividit en quatre parts que coincideixen amb les quatre pantalles que tindrà el dashboard i que s'explicaran més endavant. A diferència del Flow 1 explicat anteriorment, es farà una descripció general del funcionament del flow i dels nodes principals en comptes d'explicar tots els nodes individualment. Això és degut al fet que els flows d'elaboració del dashboard tenen una gran quantitat de nodes que influeixen en el disseny o en petites funcionalitats, però que no tenen una gran importància en el funcionament general.

6.3.1 Pantalla 1. Inici

La primera pantalla del dashboard consisteix en una introducció al servidor web on es farà una petita explicació de què és i com funciona el dish Stirling i de què és el que es trobarà a la web. Aquesta pantalla és especialment interessant perquè la idea és que el dashboard pugui ser accessible per a qualsevol persona, fet que fa imprescindible que s'afegeixi una definició del sistema dish Stirling. A l'apartat 7.1 es pot veure el resultat del flow que s'explicarà. A continuació es veu una captura (Fig.30) amb la part del flow dedicada a la pantalla 1 del dashboard:

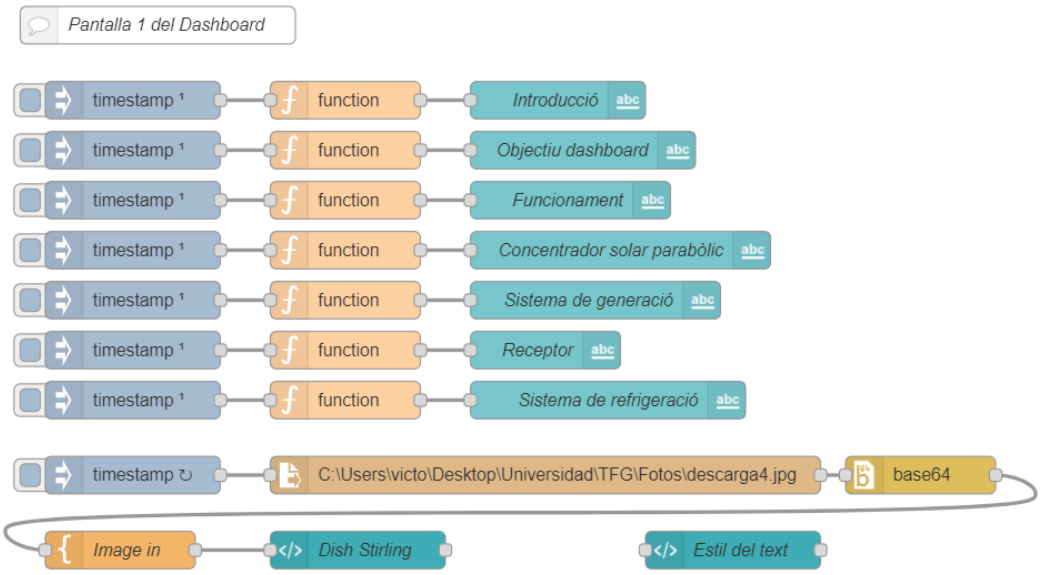
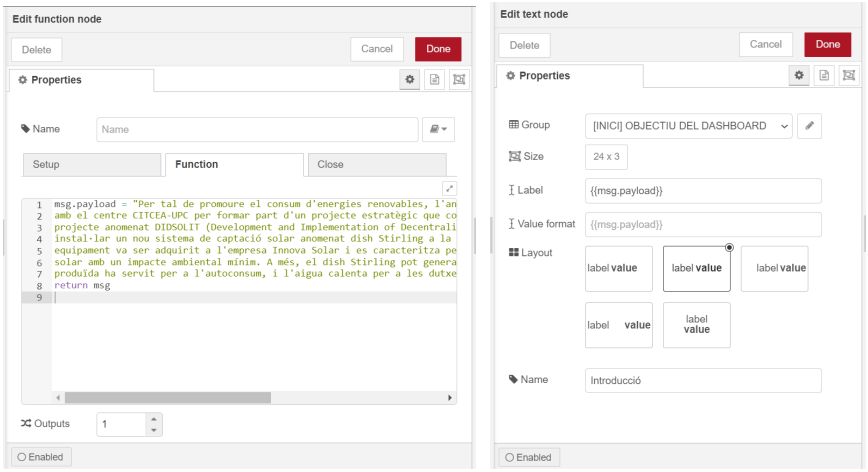


Figura 30: Flow per l'elaboració de la pantalla 1 del Dashboard [Font pròpia]

D'aquest flow es poden destacar 3 parts diferents. A la part superior es poden veure 7 línies formades cadascuna per tres nodes: un node *Inject*, un node *Function* i un node *text*. Aquesta part es dedica a introduir informació en forma de text al dashboard. Els nodes *inject* tenen la funció de fer que aparegui el text quan s'obre el dashboard, per això s'han configurat de manera que iniciïn el missatge instantàniament quan s'executi el flow. Als nodes *function* s'ha introduït el text que es vol que aparegui al dashboard, ja que resultava més còmode que escriureho al node *text*. Finalment, el node *text* és l'encarregat de fer que el text aparegui a la web. Seguidament es mostra una fotografia (Fig. 31) on es pot veure el node *function* (Fig. 31a) i el node *text* (Fig. 31b) anomenat "Introducció" perquè serveixi com a exemple de tots els nodes d'aquesta primera part.



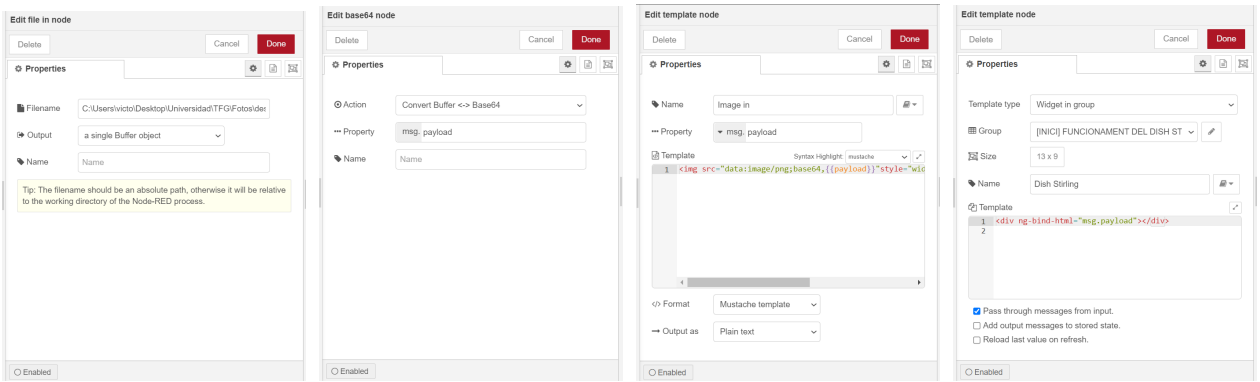
(a) Node function amb el text d'introducció

(b) Configuració del node text anomenat "Introducció"

Figura 31: Configuració dels nodes change [Font pròpia]

Com es veu a la imatge, al node *function* es defineix el msg.payload com el text que es vol que aparegui. Després, el missatge arriba al node *text* i a l'apartat "Label" s'indica que es vol que el que aparegui sigui el msg.payload. Aquest node text és un node de la llibreria dashboard on hi ha dos apartats per escriure text (l'apartat "Label" i l'apartat "Value format"). Principalment està pensat per què a la secció "Label" s'escrigui el nom d'una variable i a la secció "Value format" el valor d'aquesta variable. Tot i això, com en el cas d'aquest dashboard només es volia que aparegui text, s'ha indicat a l'apartat "Label" el text posant "msg.payload" i s'ha deixat l'altre espai buit. A més, a l'apartat "Layout" es pot escollir de quina manera apareixeran els textos introduïts.

La segona part d'aquest flow es veu a la part inferior de la figura 30 i té la funció d'incloure una imatge al dashboard. El primer node d'aquesta part és un node *inject* que inicialitza el missatge perquè s'agregui la imatge a la web. Després apareixen 4 nodes nous: un node *file in*, un node *base64*, un node *template* (llibreria function) i un node *template* (llibreria dashboard).



(a) Node file in

(b) Node base64

(c) Node template (llibreria function)

(d) Node template (llibreria dashboard)

Figura 32: Nodes de la part 2 del flow de la pantalla 1 del dashboard [Font pròpia]

- El node ***file in*** (Fig. 32a) permet accedir a un arxiu de l'ordinador. Per fer-ho s'ha d'afegir la ruta a l'apartat "Filename". Pel present treball, s'ha introduït una imatge d'un dish Stirling prèviament descarregada a l'ordinador. A l'apartat "Output" s'ha d'escollir el format del missatge que surt del node.

- El node ***base64*** (Fig. 32b) permet convertir una imatge al format base64, format que serà necessari per afegir la imatge al dashboard. Només cal indicar a l'apartat "Action" quin és l'objecte que arriba al node per convertir-lo a format base64.

- El node ***template*** de la llibreria function s'ha utilitzat per donar format a la imatge i fa que aquesta es trobi al msg.payload. Per fer-ho s'ha utilitzat la següent línia de codi: "**". Amb aquest codi es dóna mida a la imatge (700 d'alçada i 435 d'amplada) i es fa que estigui centrada.

- El node ***template*** de la llibreria dashboard s'ha utilitzat per què la imatge (introduïda al msg.payload) aparegui a la web amb el codi: "*<div ng-bind-html="msg.payload» *".

Finalment, l'última part del flow de la pantalla 1 del dashboard (Fig. 30) consisteix en un node *template* de la llibreria dashboard anomenat "Estil de text" utilitzat per donar format al text de la web (Fig. 33). Com es vol fer ús del node per modificar l'estil del text, s'ha de començar el codi escrivint *<style>* i finalitzar-lo amb *</style>*. Primer s'ha d'indicar el nom de la pantalla i del grup del dashboard al qual es vol aplicar el codi. Per fer que el text aparegui justificat, s'utilitza el codi: "*text-align: JUSTIFY*". A més, es pot canviar la mida de la lletra amb la comanda *font-size* i canviar els marges amb la comanda *margin*. En el cas de la imatge de la dreta, només s'ha modificat el marge superior i per tant la comanda és *margin-top*.

```
<style>
#INICI_OBJECTIU_DEL_DASHBOARD {
    text-align: JUSTIFY;
    font-size: 17px;
    margin-top:15px;
}

#INICI_FUNCIONAMENT_DEL_DISH_STIRLING{
    text-align: JUSTIFY;
    font-size: 17px;
    margin-top:30px;
}

}
</style>
```

Figura 33: Codi del node template [Font pròpia]

6.3.2 Pantalla 2. Variables

La segona pantalla del dashboard consisteix en la visualització de l'històric de les variables del dish Stirling. Per tant, la pantalla del dashboard mostrarà un gràfic i un seguit de filtres amb els quals poder escollir què es vol visualitzar a aquest gràfic. A l'apartat 7.2 es pot veure el resultat del flow que s'explicarà. A continuació es veu una captura (Fig. 34) amb la part del flow dedicada a la pantalla 2 del dashboard:

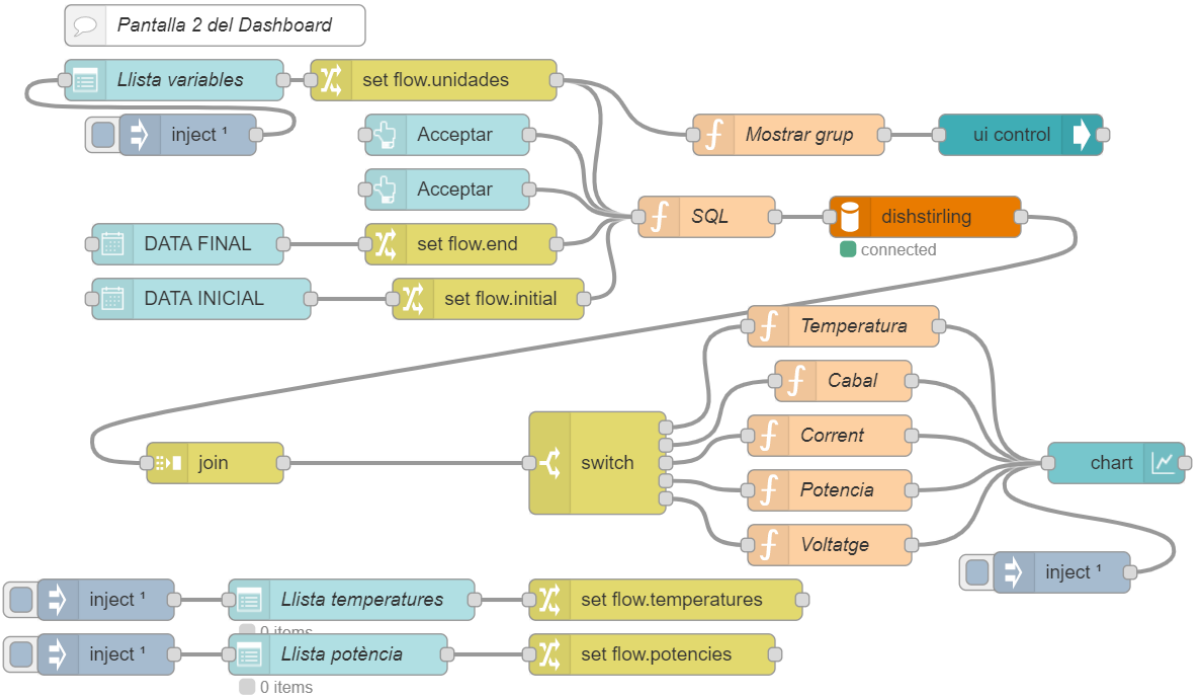


Figura 34: Flow per l'elaboració de la pantalla 2 del Dashboard [Font pròpia]

El node principal de tot el flow és el node *function* anomenat SQL que es troba a la part central de la imatge 34, ja que és el node encarregat de seleccionar les dades que s'agafaran de la base de dades. A aquest node arriben 5 branques diferents provinents de 5 nodes. En el moment en què qualsevol d'aquests nodes envia un msg.payload la funció selecciona les dades i aquestes es mostren al gràfic. Per tant, hi ha 5 maneres de poder mostrar al gràfic les dades al dashboard. Començant per la part superior, veiem un node *dropdown* anomenat "Llista variables" que fica al dashboard un desplegable amb totes les variables. En aquesta llista hi ha 5 elements: temperatura, cabal, corrent, potència i voltatge. Escollint algun d'aquests elements es mostrarà el gràfic corresponent a aquestes variables. D'aquests 5 elements, hi ha 2 dins dels quals hi ha més d'una variable; és el cas de la temperatura i la potència. Com es pot veure a la part inferior de la imatge, hi ha dos nodes *dropdown*: un amb la llista de variables de temperatura i un altre amb la llista de potències. Això permetrà que quan a la llista de variables se seleccioni la temperatura o la potència es pugui escollir quina temperatura o potència es vol afegir al gràfic. A diferència del primer *dropdown* esmentat, aquests permeten la selecció múltiple per si es vol veure al gràfic més d'una temperatura o potència a l'hora. Seguit dels nodes *dropdown*, hi ha dos nodes *change* per emmagatzemar la temperatura i la potència escollida a les propietats flow.temperatures i flow.potencies, respectivament.

Seguit del node anomenat "Llista variables" també hi ha un node *change* per emmagatzemar l'element de la llista seleccionat a la propietat *flow.unidades*. En el moment de seleccionar l'element de la llista s'envia el missatge, el qual arriba al node "SQL" (després de passar pel node *change*) i permet afegir al gràfic la variable escollida. Del node *change* surt una altra branca dirigida cap a un node *function* anomenat "Mostrar grup". El contingut d'aquesta funció es pot veure a la imatge següent (Fig. 35).

Figura 35: Contingut del node function anomenat "Mostrar grup" [Font pròpia]

Abans d'explicar el codi del node function, s'ha de tenir en compte que el missatge que surt d'aquest node va dirigit a un node *ui_control*. Aquest, és un node que permet canviar la distribució del dashboard en funció del msg.payload que li arriba. Com s'ha comentat anteriorment, dins de la temperatura i la potència hi ha moltes variables. La intenció d'incloure aquest node és fer que quan es seleccioni a la llista de variables l'element temperatura o l'element potència aparegui un nou desplegable on poder seleccionar les temperatures o les potències, respectivament.

A la primera línia del codi (Fig. 35) s'indica que si la propietat *flow.unidades* és igual a "Temperatura" s'enviï un msg.payload al node *ui_control* perquè mostri el desplegable de temperatures (aquest es troba dins del grup "Seleccionar temperatures" de la pantalla del dashboard "VARIABLES") i perquè mantingui amagat el desplegable per seleccionar les potències (aquest es troba dins del grup "Seleccionar potències" de la pantalla del dashboard "VARIABLES"). En cas que la propietat *flow.unidades* sigui igual a "Potència" es faria a la inversa i si la propietat *flow.unidades* no és igual a cap d'aquests dos strings, es mantindran amagats els dos desplegables.

Una altra forma d'iniciar el flow és utilitzar els nodes *button* anomenats "Acceptar" mostrats a la part superior esquerra de la imatge 34. Aquests botons apareixeran al dashboard juntament amb les llistes desplegables de temperatura i potència que s'han comentat anteriorment. Sota d'aquests nodes hi ha dos nodes *date picker* seguits de dos nodes *change*. Aquests nodes s'utilitzaran per indicar la data inicial i la data final que definirà l'interval de temps que es vol veure al gràfic. Aquesta informació s'emmagatzemarà respectivament a les propietats *flow.initial* i *flow.end*.

A continuació s'explicarà el node central del flow, el node function SQL. A la imatge següent (Fig. 36) es pot veure el contingut del codi.

```
var Week  = 604800000 ; //7 Days
var Day   =  86400000 ; // 1 Days
var d = new Date();
var epoch = d.getTime();
var fromdate = flow.get("initial");
var enddate = flow.get("end") + Day;
var output = [];

if (isNaN(enddate)) {
    enddate = epoch
}

output.push({ topic: "SELECT * FROM trinum WHERE epoch >= " + fromdate + " AND epoch <= " + enddate });

output[output.length-1].complete=true;

return [ output ];
```

Figura 36: Contingut del node function anomenat "SQL" (Pantalla 2) [Font pròpia]

A les primeres dues línies del codi es calcula el temps d'una setmana i d'un dia en mil·lisegons per poder-lo manipular amb l'epoch. A la línia 3 es defineix la variable "d" com una nova data a la qual s'atorga el moment actual a la variable "epoch". A les línies 5 i 6, es defineixen les variables "fromdate" i "enddate" agafant el valor emmagatzemat a les propietats *flow.initial* i *flow.end*, respectivament. A la variable "enddate" se li ha afegit un dia perquè si no s'afegeix quan se selecciona la data final el gràfic només mostraria les variables fins al dia escollit sense incloure-ho. A la línia 7 es defineix una llista buida amb el nom "output". Més endavant, es crea una condició perquè si l'enddate no té cap valor, s'agafi com a valor la data actual. Finalment, s'envia el missatge perquè se seleccionin les variables de la base de dades que es troben entre les dates indicades.

Continuant la figura 34 s'observa que després de passar pel node SQL el missatge arriba al node *join* el qual s'encarrega de convertir les dades obtingudes de la base de dades en un array que facilita la manipulació d'aquestes dades. A continuació, s'arriba al node *switch*, el qual divideix el missatge en funció de quin element de la llista de variables s'ha seleccionat. Per tant, el flow es divideix en 5 branques que arriben a 5 nodes *function*, un per cada variable de la llista. A la imatge inferior (Fig. 37) es mostra el codi del node *function* Potència. Només s'explicarà aquest node, ja que el codi dels altres quatre és igual canviant únicament el nom de les variables.

```
1  var msg2 = [];
2  var llista = flow.get("potencies");
3  var valors = []
4  if (msg.payload[0].length == 0 || llista.length == 0){
5      msg2=[];
6  }
7  else {
8      for (var i=0; i<msg.payload.length; i++) {
9          var output = [];
10         var output1 = [];
11         for (var j=0; j<msg.payload[i].length; j++) {
12             if(llista.includes("E_POWER")){
13                 output.push({"x": msg.payload[i][j].epoch, "y": msg.payload[i][j].E_POWER});
14             }
15             if(llista.includes("P_TERM")){
16                 output1.push({"x": msg.payload[i][j].epoch, "y": msg.payload[i][j].P_TERM});
17             }
18         }
19         if(llista.includes("E_POWER")){
20             valors.push(output);
21         }
22         if(llista.includes("P_TERM")){
23             valors.push(output1);
24         }
25         msg2.push({ series: flow.get("potencies"), data: valors});
26         //msg2.push({ key: "test", values : output});
27     }
28 }
29 msg.payload=msg2;
30 return msg;
```

Figura 37: Contingut del node function anomenat "Potència" [Font pròpia]

A les primeres tres línies del codi es defineixen dues variables buides: "valors" i "msg2"; i una variable anomenada "llista" que és igual al valor de la propietat *flow.potencia* (element seleccionat de la llista de potències). A partir de la línia 4 s'estableix la condició que si la variable llista està buida o si el missatge que arriba no té cap dada s'enviï un missatge buit i, per tant, el gràfic estarà buit. En cas de no complir-se aquesta condició, es recorrerà l'array que li arriba amb les dades del dish Stirling mitjançant un *for*. Dins d'aquest bucle, s'introdueix a la variable output el valor de l'epoch i de la potència de cada línia de l'array. Per fer-ho, s'utilitza la codificació:

output.push("x":msg.payload[i][j].epoch, "y":msg.payload[i][j].<nom de la potència>

Aquests valors de l'output després s'afegeixen a la llista "valors", ja que el missatge amb les dades que ha d'arribar al node *chart* és una llista de llistes. El missatge que ha d'arribar al node *chart* ha de seguir la següent notació:

({series: <nom de les potències>, data: <llistat del valor de les potències i el seu epoch>})

Finalment, el missatge arriba a l'últim node del flow que és el node *chart*. A la figura 38 es mostra la configuració del node del flow. Com es veu a l'apartat "type" s'ha escollit un gràfic de línies i al "X-axis Label" s'ha de definir el format com apareixerà les dates de l'eix x. En el nostre cas, les dates apareixeran en format "Dia HH:MM". També es pot escollir el valor mínim i màxim de l'eix y. En el cas del flow actual aquest apartat s'ha deixat en blanc perquè el gràfic ajusti l'eix automàticament. A més, es pot escollir entre mostrar o no la llegenda del gràfic i els colors de les línies del gràfic. Tot i que no es veu a la imatge també hi ha altres apartats: està l'apartat "Name" com a tots els nodes i l'apartat "Group" com a tots els nodes de la llibreria dashboard. Finalment, a l'apartat "blank label" s'ha de definir si es vol que el gràfic mostri algun text quan no arribi cap dada (quan arribi un missatge buit: []). A aquest blank label s'ha escrit el següent missatge: "No hi ha dades o els filtres seleccionats no són vàlids", ja que el gràfic estarà buit quan no hi hagi dades entre les dates seleccionades, quan s'han seleccionat les dates incorrectament (per exemple, escollir una data inicial posterior a la data final) o quan no s'ha seleccionat la variable que es vol mostrar al gràfic.

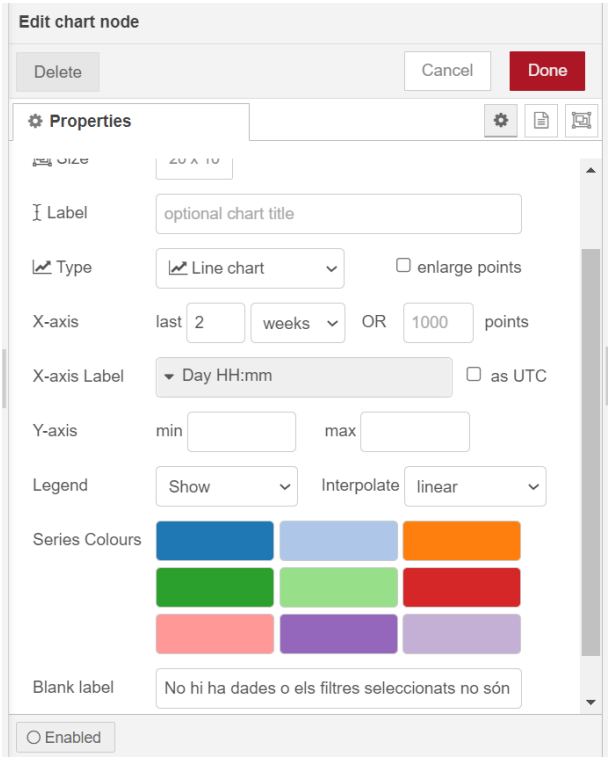


Figura 38: Configuració del Node chart [Font pròpia]

Cal destacar la incorporació de 4 nodes *inject* al flow. Aquests nodes s'han col·locat abans dels nodes *dropdown* i del node *chart* per reiniciar els seus valors cada cop que s'obre la web. Per fer-ho el node *inject* envia l'element "[]" al msg.payload. No és necessari aplicar els nodes *inject* abans dels nodes *date picker* perquè per defecte apareix la data actual.

6.3.3 Pantalla 3. Generació energètica

La tercera pantalla del dashboard consisteix en la visualització de l'històric de la generació d'energia (tant elèctrica com tèrmica) del dish Stirling. Per tant, la pantalla del dashboard mostrarà 4 gràfics per poder veure la generació d'energia: generació diària, generació setmanal, generació mensual i generació anual. A més, es podrà escollir entre veure l'energia elèctrica o l'energia tèrmica. A l'apartat 7.3 es pot veure el resultat del flow que s'explicarà. A continuació es veu una captura (Fig. 39) amb la part del flow dedicada a la pantalla 3 del dashboard:

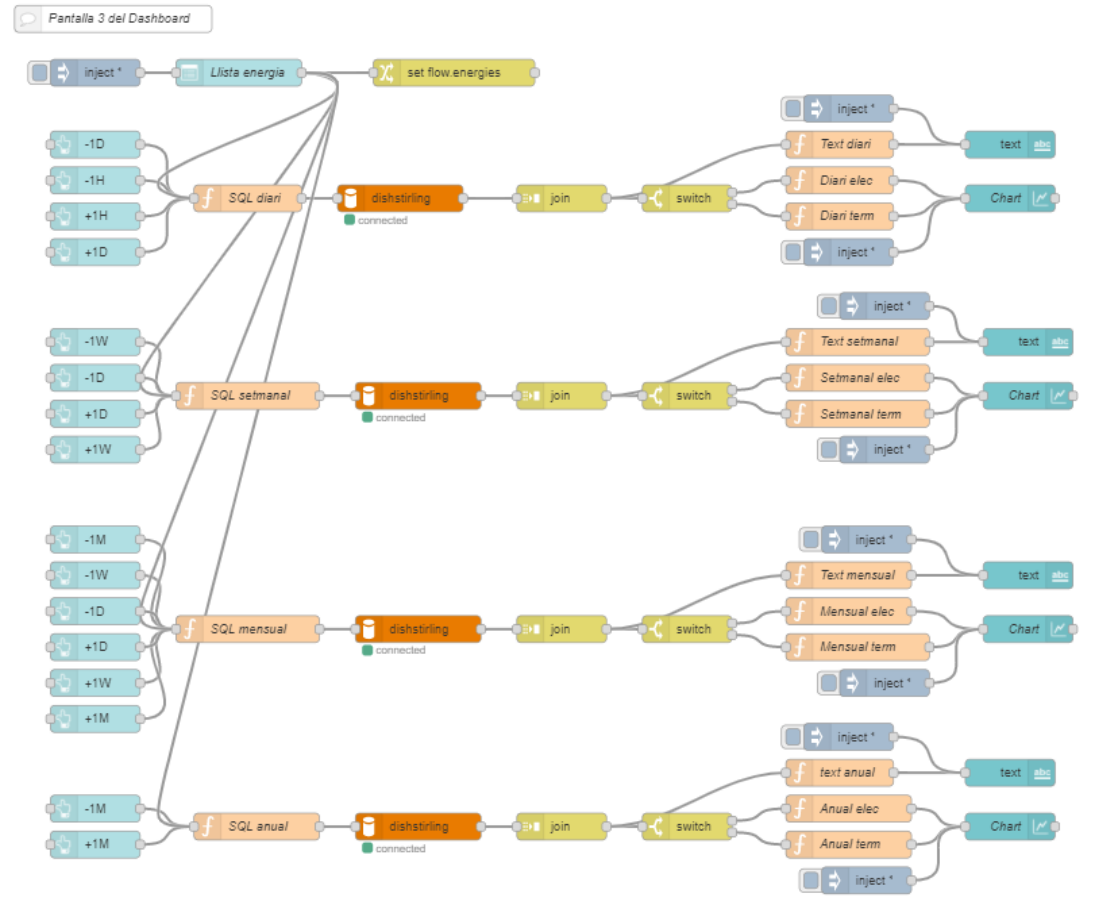


Figura 39: Flow per l'elaboració de la pantalla 3 del Dashboard [Font pròpia]

Al flow de la imatge (Fig. 39) destaquen quatre línies que coincideixen amb els quatre gràfics de la generació d'energia. A la part superior apareix un node *dropdown* anomenat "Llista energia" per poder escollir al dashboard entre les dues energies i s'emmagatzemarà la variable escollida a la propietat *flow.energies* amb un node *change*. A l'esquerra del flow apareixen molts nodes *button* que s'utilitzaran per avançar (+) o endarrerir (-) els gràfics un determinat període de temps: una hora (1H), un dia (1D), una setmana (1W) o un mes (1M). Cadascun dels gràfics es podran desplaçar només amb alguns d'aquests polsadors, ja que, per exemple, no tindria sentit desplaçar una hora el gràfic de la generació d'energia anual. Per explicar el funcionament del flow, s'agafarà com a exemple la línia superior, encarregada de fer el gràfic de generació diària. El flow comença amb els polsadors perquè cada cop que es polsi un s'actualitzi el gràfic. Després, el missatge arriba a un node *function* anomenat "SQL diari".

El codi d'aquest node es pot veure a la figura 40. Les primeres dues variables estan destinades a definir un període de temps (hora i dia) en format epoch (en mil·lisegons). De la línia 9 a la 16 s'estableix que en cas que les variables enddate i fromdate no estiguin definides siguin igual al temps actual i al temps actual menys un dia, respectivament. De la línia 18 a la línia 43 del codi s'estableix el funcionament dels polsadors. Per exemple, quan es polsa el botó -1H (el qual envia un msg.topic = minus1h) tant la variable fromdate com la variable enddate s'endarrereix un dia. Finalment, s'envia la sentència perquè la base de dades retorni les variables entre els instants establerts.

```
1  var p_1h = 1000*60*60 ; // 1 Hora
2  var p_1d = 1000*60*60*24 ; // 1 Dia
3  var d = new Date();
4  var current = d.getTime();
5  var output = [];
6  var enddate = 0;
7  var fromdate = 0;
8
9  fromdate = context.get("fromdate");
10 if (fromdate===undefined) {
11     fromdate = current-p_1d;
12 }
13 enddate = context.get("enddate");
14 if (enddate===undefined) {
15     enddate = current;
16 }
17
18 switch(msg.topic) {
19     case "minus1h":
20         fromdate = fromdate-p_1h;
21         enddate = enddate-p_1h;
22         context.set("fromdate",fromdate);
23         context.set("enddate",enddate);
24         break;
25     case "plus1h":
26         fromdate = fromdate+p_1h;
27         enddate = enddate+p_1h;
28         context.set("fromdate",fromdate);
29         context.set("enddate",enddate);
30         break;
31     case "minus1d":
32         fromdate = fromdate-p_1d;
33         enddate = enddate-p_1d;
34         context.set("fromdate",fromdate);
35         context.set("enddate",enddate);
36         break;
37     case "plus1d":
38         fromdate = fromdate+p_1d;
39         enddate = enddate+p_1d;
40         context.set("fromdate",fromdate);
41         context.set("enddate",enddate);
42         break;
43 }
44
45 output.push({ topic: "SELECT * FROM trinum WHERE epoch >= " + fromdate + " AND epoch <= " + enddate });
46
47 output[output.length-1].complete=true;
48
49 return [ output ];
```

Figura 40: Contingut del node function anomenat "SQL diari" [Font pròpia]

El missatge del node *function* arriba fins al node *mysql* d'on s'extreuen les variables i es passen a un format adient (un array) amb el node *join*. Després s'arriba a un node *switch* on el missatge es divideix en dues branques: una si se selecciona l'energia elèctrica i altra per l'energia tèrmica. Ara s'arriba a dos node *function* els quals contenen el mateix codi només canviant el nom de la variable. Com a exemple, es mostra el codi del node "Diari elec" (Fig. 41). S'ha de tenir en compte que el gràfic al qual arribarà el missatge serà un gràfic de barres. En aquest tipus de gràfics, al node ha d'arribar una array amb tres claus: series (on s'indica la variable), data (on apareixen els valors) i labels (on s'afegeix el que sortirà a l'eix horitzontal). Això es mostra de la línia 1 a la línia 5 del codi. Fins a la línia 11 es defineixen les variables que s'utilitzaran més endavant, destaca la variable "l" que emmagatzemarà l'últim valor d'energia i la variable "valor" que emmagatzemarà la diferència d'energia entre dos instants (l'energia generada entre aquests dos instants). Cal recordar que el valor de l'energia que apareix a la base de dades és el valor acumulat (a l'instant 0 l'energia és 0 i a l'instant n és tota l'energia que s'hagi generat des de l'instant 0 fins a l'instant n). De la línia 12 a la línia 32 s'estableix el bucle per recórrer tota l'array de valors que arriba al node i agafar els valors de les variables en el format adient per enviar-ho al node *chart*. El primer cop que es recorri l'array la variable l serà 0 i, per tant, s'emmagatzemarà els valors de l'energia i l'epoch del primer instant a les variables l i d1, respectivament (línies 16-19). Com que el gràfic calcularà la generació d'energia a cada hora del dia, a les línies 20-22 es comprovarà si l'hora de l'instant recorregut és la mateixa que l'hora de l'últim instant emmagatzemat, si és així passarà a recórrer el següent instant. Si l'hora no

coincideix, es calcularà la variable valor (restant l'energia de l'actual instant amb la de l'últim valor emmagatzemat) per introduir-la a l'apartat data (línia 24 i 26) de la variable m i l'apartat label s'afegirà les hores entre les quals s'ha generat aquesta energia: hora1-hora2 (línia 27). A més, s'actualitzen els valors de les variables l i d1 amb els valors d'aquest instant. Recorrent tota l'array es calcularà la generació a cada hora. Aquesta manera de calcular l'energia té el problema de que no calcularia l'energia de l'última hora i això s'ha corregit amb les línies de la 34 a la 44 on es calcula la generació entre l'últim instant emmagatzemat i l'últim instant del array. Finalment, s'envia la variable m (que conté tota la informació ordenada) al node *chart*.

```
1  var m={
2      "series":["Energia elèctrica"],
3      "data":[[]],
4      "labels":[]
5  };
6  var l = 0;
7  var valor = 0;
8  var long = msg.payload[0].length-1;
9  var d = new Date();
10 var d1 = new Date();
11 var d2 = new Date();
12 if (msg.payload[0].length>0) {
13     for (var i=0; i<msg.payload.length; i++) {
14         for (var j=0; j<msg.payload[i].length; j++) {
15             d.setTime(msg.payload[i][j].epoch);
16             if (l == 0){
17                 l = msg.payload[i][j].E_ENERGY;
18                 d1.setTime(msg.payload[i][j].epoch);
19             }
20             else if (d.getHours()==d1.getHours()){
21                 //pass
22             }
23             else{
24                 valor = (msg.payload[i][j].E_ENERGY - l);
25                 l = msg.payload[i][j].E_ENERGY;
26                 m.data[0].push(valor);
27                 m.labels.push(d1.getHours()+"-"+d.getHours());
28                 d1.setTime(msg.payload[i][j].epoch);
29             }
30         }
31     }
32 }
33
34 if(msg.payload[0][long].epoch == d1){
35     //pass
36 }
37
38 else{
39     valor = msg.payload[0][long].E_ENERGY - l;
40     m.data[0].push(valor);
41     d.setTime(msg.payload[0][long].epoch);
42     d2.setTime(msg.payload[0][long].epoch+1000*60*60);
43     m.labels.push(d.getHours()+"-"+d2.getHours());
44 }
45
46 return {payload:[m]};
```

Figura 41: Contingut del node function anomenat "Diari elec" [Font pròpia]

Continuant amb el flow de la imatge 39, el missatge passa del node *function* al node *chart*, la configuració del qual coincideix amb la de la imatge 38 però seleccionant bar chart a l'apartat "Type". A la tercera pantalla dels dashboard, juntament amb cada gràfic s'inclourà un text per indicar la generació total entre els dos instants que limiten el període del gràfic. Per fer-ho s'ha afegit un node *function* i un node *text* al flow de la figura 39. Com a exemple, s'explicarà el codi del node *function* anomenat "Text diari" que es pot veure a la imatge (Fig. 42) següent:

```
1  var lista = msg.payload;
2  var long = lista[0].length-1;
3  var tipus = "";
4  var total = 0;
5  var elec1 = lista[0][0].E_ENERGY;
6  var elec2 = lista[0][long].E_ENERGY;
7  var totalelec = elec2 - elec1;
8  var term1 = lista[0][0].E_TERM;
9  var term2 = lista[0][long].E_TERM;
10 var totalterm = term2 - term1;
11 var epoch1 = lista[0][0].epoch;
12 var epoch2 = lista[0][long].epoch;
13 var d1 = new Date();
14 var d2 = new Date();
15 d1.setTime(epoch1);
16 d2.setTime(epoch2);
17 var hora2 = d2.getHours() + 1;
18 var temps1 = d1.getDate() + "/" + d1.getMonth() + "/" + d1.getFullYear() + " a les " +  d1.getHours() + " hores";
19 var temps2 = d2.getDate() + "/" + d2.getMonth() + "/" + d2.getFullYear() + " a les " +  hora2 + " hores";
20 if (flow.get("energies") == "E_ENERGY"){
21     tipus = "elèctrica";
22     total = Math.round(totalelec*10000)/10000;
23 }
24 else if (flow.get("energies") == "E_TERM"){
25     tipus = "tèrmica";
26     total = Math.round(totalterm*10000)/10000;
27 }
28 msg.payload = "L'energia " + tipus + " generada des del " + temps1 + " fins el " + temps2 + " és de " + total + " kWh";
29 return msg;
30
```

Figura 42: Contingut del node function anomenat "Text diari" [Font pròpia]

S'ha de tenir en compte que aquest node function està unit amb el node *join* i que, per tant, l'arribarà un array amb els valors entre els instants del gràfic. A les variables del codi "elec1" i "elec2" es calcula el primer i l'últim valor d'energia de l'array respectivament per a després calcular la generació total restant aquests dos valors a la variable "totalelec". El mateix es fa amb l'energia tèrmica i també es guarden el primer i l'últim valor de l'epoch a les variables "epoch1" i "epoch2". A les variables "temps1" i "temps2" es posen aquests dos instants en un format determinat: "DD/MM/YYY a les HH hores". La variable tipus serà igual a "elèctrica" o "tèrmica" en funció de l'energia escollida i la variable "total" serà l'arrodoniment a 4 decimals del total generat. Finalment, a la línia 28 s'estableix al msg.payload la comanda que s'enviarà perquè aparegui el text al dashboard. Al node *text* només s'haurà de posar msg.payload a l'apartat "Label".

Cal destacar que s'han afegit nodes *inject* abans de tot els nodes *chart*, nodes *text* i al node *dropdown* per tal d'inicialitzar el seu valor quan s'obre el dashboard.

6.3.4 Pantalla 4. Consulta de variables

La quarta pantalla del dashboard consisteix en la visualització dels valors de les variables del dish Stirling en un moment indicat. Per tant, la pantalla del dashboard mostrarà totes les variables amb el seu corresponent valor i una línia de filtres per visualitzar les dades d'un dia, hora i minuts seleccionats. A l'apartat 7.4 es pot veure el resultat del flow que s'explicarà. A continuació es veu una captura (Fig. 43) amb la part del flow dedicada a la pantalla 4 del dashboard:

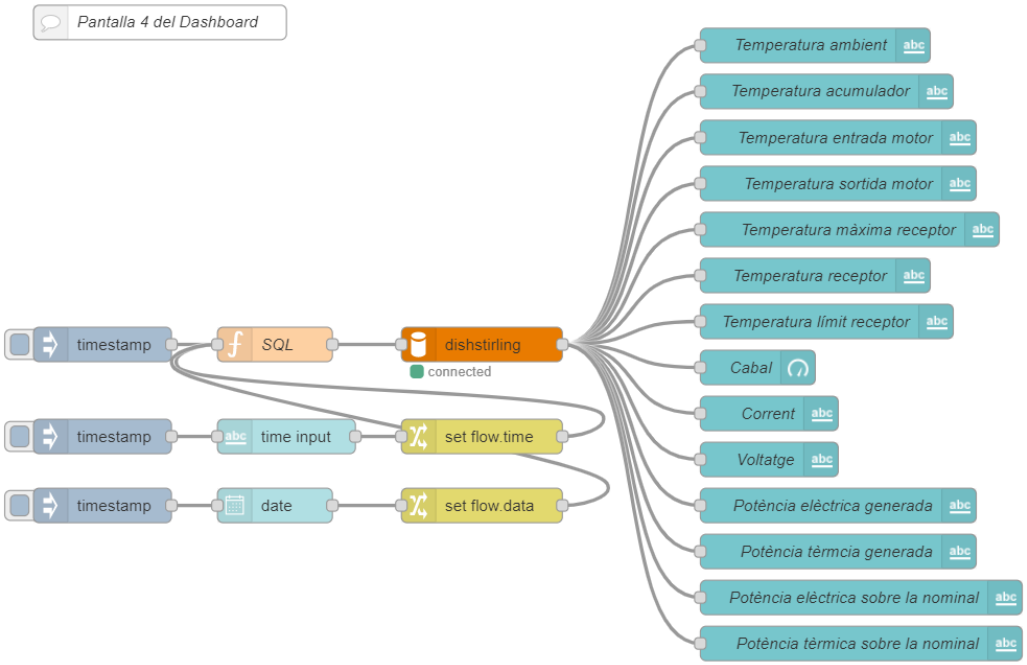


Figura 43: Flow per l'elaboració de la pantalla 4 del Dashboard [Font pròpia]

El flow mostrat a la imatge té tres inicis que coincideixen amb tres nodes *inject*, la funció dels quals és únicament la d'inicialitzar el missatge quan s'executa el flow. A la part esquerra de la figura 43 es mostra un node *date picker* per poder escollir el dia en què es vol consultar el valor de les variables. El valor d'aquest node s'emmagatzemarà a la propietat *flow.time* amb un node *change*. Més a dalt, hi ha un node *text input* la configuració del qual es pot veure a la imatge dreta (Fig. 44). Aquest node permet afegir al dashboard un camp d'entrada de text. Com a qualsevol node trobem l'apartat "Name" i com a tots els nodes de la llibreria dashboard apareix l'apartat "Group". L'apartat més important de la configuració d'aquest node és l'apartat "Mode" on es pot escollir la funció del node d'entre les següents: text input, mail address, password, number, telephone input, color picker, time picker, week picker i month picker. En el nostre cas, s'ha escollit l'opció time picker per poder seleccionar al dashboard l'hora i minuts del moment en què es volen veure les dades. Aquest valor s'enviarà al msg.payload i com es pot veure a la figura superior (Fig. 43) s'emmagatzemarà a la propietat *flow.data* amb un node *change*.

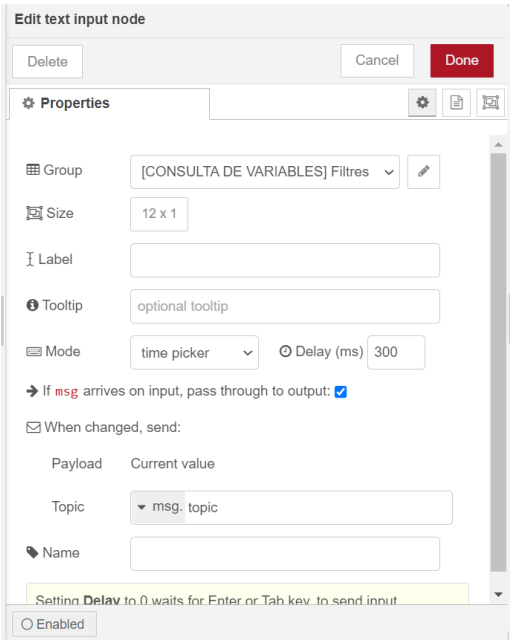


Figura 44: Configuració del Node text input [Font pròpia]

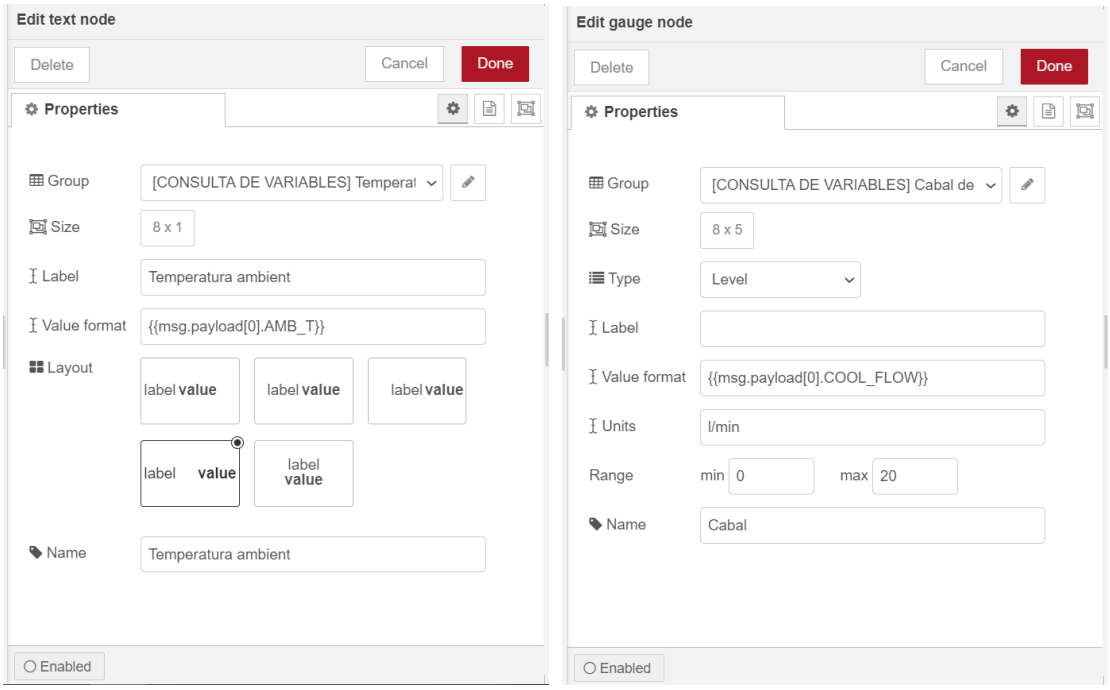
Els dos nodes *change* arriben a un node *Function*, el contingut del qual es troba a la imatge següent (Fig. 45):

```
1  var data = flow.get("data") - 2*60*60*1000;
2  var temps = flow.get("time");
3  var epoch = data + temps;
4  var output = [];
5
6  output.push({ topic: "SELECT *, ABS(" + epoch + " - epoch) AS X FROM trinum ORDER BY X LIMIT 1"});
7
8  output[output.length-1].complete=true;
9
10 return [ output ];
```

Figura 45: Contingut del node function anomenat "SQL" (Pantalla 4) [Font pròpia]

A les primeres 4 línies del codi es defineixen les variables. La primera és la variable "data" que és el dia que s'esculli al node *date picker*. Cal destacar que aquest node retorna un epoch de les 00:00 del dia escollit i que a aquest valor se li han restat dues hores (2*60*60*1000 mil·lisegons) per adaptar el valor a l'hora peninsular espanyola. La variable "temps" es defineix com el valor escollit al node *text input*. Finalment, la variable "epoch" és la suma de les dues variables anteriors per aconseguir l'epoch exacte del dia, hora i minuts escollits i la variable "output" és una llista buida. A la línia 6 es defineix el missatge que s'enviarà a la base de dades en el msg.topic. S'ha de tenir en compte que com a la base de dades s'inclouen valors cada cert temps (s'ha configurat que s'afegeixin cada 10 minuts tot i que es podria modificar) és impossible que l'epoch del moment escollit (dia, hora i minuts) coincideixi amb l'epoch d'un dels valors de la base de dades. Per aquest motiu, l'objectiu serà retornar les variables del moment més proper a l'escollit als filtres del dashboard. Per fer-ho, es defineix amb el nom "X" al valor absolut de la resta entre la variable epoch (el valor de temps escollit) i l'epoch de cada línia de valors de la base de dades. Això es fa amb la comanda: "SELECT *, ABS(<var epoch> - epoch) AS X FROM <nom de la taula>". A més, es demana que s'ordeni la base de dades de manera ascendent segons el valor de X (el valor absolut de la resta) i que retorni el primer valor afegint a la comanda el següent: "ORDER BY X LIMIT 1". D'aquesta manera es retorna la línia de valors de la base de dades amb l'epoch més proper al moment demanat al dashboard.

Continuant amb la figura 43, del node *mysql* surt un missatge amb el valor de les variables i arriba a diversos nodes *text*, un per a cada variable, i a un node *gauge* per la variable cabal. Aquests nodes permeten mostrar text o, en el cas del node *gauge*, elements de calibratge al dashboard. Es mostrarà a continuació (Fig. 46) la configuració del node anomenat "Temperatura ambient" com a exemple de tots els nodes *text* i la configuració del node anomenat "Cabal".



(a) Node text (b) Node gauge

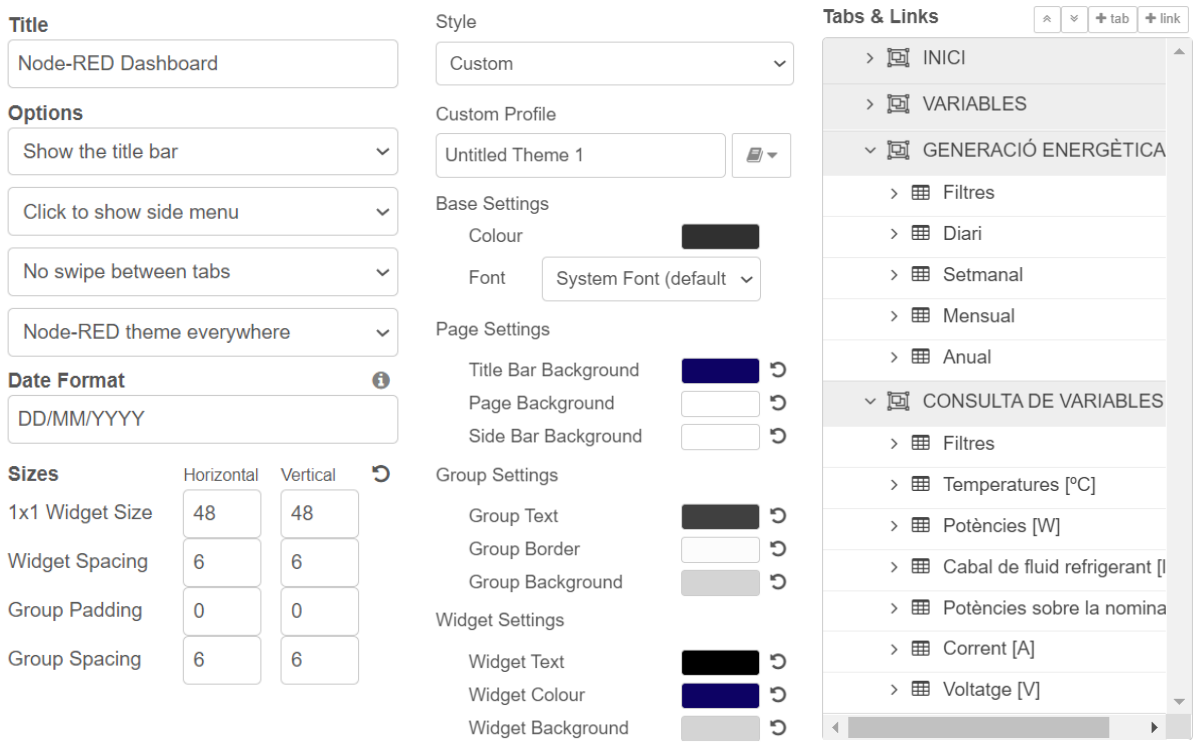
Figura 46: Configuració dels nodes text i gauge (Pantalla 4) [Font pròpia]

En ambdós nodes, per obtenir el valor desitjat del missatge que arriba al node s'ha d'introduir a l'apartat "Value format" la comanda: {{msg.payload[0].<nom de la variable>}}. Pel que fa al node *text* (Fig. 46a), cal destacar l'apartat "Label" on s'ha d'escollir el rètol que es vol que aparegui juntament amb el valor i l'apartat "Layout" on es pot escollir el format en què es vol que aparegui el rètol i el valor de la variable. Per part del node gauge (Fig. 46b), cal destacar l'apartat "type", on es pot escollir quin element es vol que mostri al dashboard. Les opcions són: Gauge (calibre), Donut, Compass (brúixola) i Level (nivell). S'ha considerat que l'opció més adient per mostrar el valor del cabal és el tipus Level. A més, s'ha de determinar el rang determinant el valor mínim i màxim (en aquest cas de 0 a 20) i les unitats de la variable (litres/minut).

7 Dashboard

Un cop finalitzada l'explicació de l'elaboració del dashboard, es mostrarà el resultat final d'aquest, indicant el seu funcionament i ensenyant el seu disseny i estètica. Previ a la realització de cada pantalla, s'ha dut a terme un *Mock-Up*, entenent per Mock-Up el prototipus del disseny final d'un dashboard. Aquest disseny previ es farà amb el software *Canva*, que és un lloc web d'eines de disseny gràfic simplificat. Durant l'explicació de cada pantalla s'afegirà una captura del mock-up i del resultat final per comparar ambdós dissenys.

Abans de fer l'elaboració del dashboard, s'ha decidit la configuració general d'aquest amb el software Node-Red (Fig. 47).



(a) Configuració general (b) Configuració cromàtica (c) Configuració de la disposició

Figura 47: Configuració general del dashboard [Font pròpia]

Com es veu a la imatge 47a, es pot escollir les següents opcions: mostrar o no la barra del títol, que es mostri el menú lateral sempre o només quan es premi, poder canviar de pantalla lliscant amb el ratolí i on aplicar la configuració cromàtica seleccionada. A més, s'ha de triar el format de les dates i la mida dels elements. A la figura 47b, es pot seleccionar la gamma cromàtica de la pàgina, els grups i els elements utilitzats. També es pot seleccionar una configuració predeterminada. Finalment, a la captura 47c, es veu la distribució de les quatre pantalles del dashboard (Inici, Variables, Generació energètica i Consulta de variables) dins de la qual hi ha els grups on s'afegiran els elements, gràfics o textos necessaris. Pressionant a cadascun dels noms de les pantalles del dashboard, apareix una opció anomenada *layout*. Amb aquesta opció, es permet organitzar la distribució dels grups del dashboard (Fig. 48). D'aquesta manera es pot decidir la mida dels elements de cada grup i l'organització d'aquests.

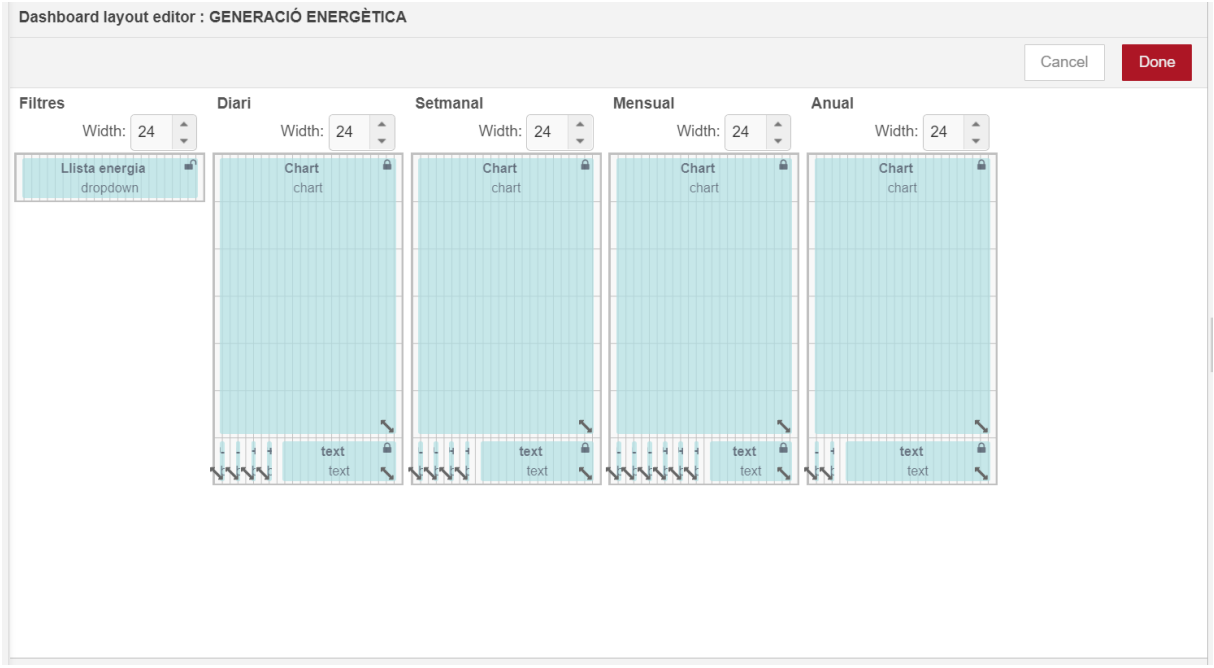


Figura 48: Layout editor de la pantalla del dashboard "Generació energètica" [Font pròpia]

A més d'aplicar la configuració general al dashboard, s'han fet servir nodes *template* al Node-Red per afegir alguns detalls i fer petites modificacions estètiques al dashboard (Fig. 49). A la imatge 49a, es veu el codi utilitzat per canviar: el color del fons de les llistes on se selecciona les variables (elements *dropdown*), el color del text que apareix quan no arriben dades als gràfics (blank label) i el color dels títols de cada grup del dashboard. D'altra banda, a la figura 49b es veu el codi necessari per afegir el logo de l'ETSEIB a la part superior dreta del dashboard.

```
<style>
   .nr-dashboard-theme md-select-menu md-option {
        background-color: #FFFFFF;
        color: #0D0264;
        background: #FFFFFF;
    }
    p.blank-label {
        color: black !important;
    }
    .nr-dashboard-theme ui-card-panel p.nr-dashboard-cardtitle {
    color: #0D0264;
    }
</style>
```

(a) Configuració de detalls estètics

```
<script id="clockScript1" type="text/javascript">
    $(function () {
        var div1 = $('<div/>');
        var logo = new Image();
        logo.src = 'https://etseib.upc.edu/logoPropi.png'
        logo.height = 45;
        div1[0].style.margin = '5px 5px 5px auto';
        div1.append(logo);
        function addToToolbar() {
            var toolbar = $('.md-toolbar-tools');
            if(!toolbar.length) return;
            toolbar.append(div1);
        }
    });
</script>
```

(b) Addició del logo de l'ETSEIB al dashboard

Figura 49: Configuració general del dashboard [Font pròpia]

7.1 Pantalla 1. Inici

La primera pantalla del dashboard consisteix en una introducció al que és el dish Stirling fent una petita explicació del seu funcionament. L'elaboració d'aquesta pantalla mitjançant el software Node-Red va ser explicada a l'apartat 6.3.1. Aquesta pantalla és l'única del dashboard que no és interactiva, ja que només conté text i una imatge. Seguidament es mostra una imatge del disseny inicial que es va fer d'aquesta pantalla:

Figura 50: Mock-Up de la pantalla 1 del dashboard [Font pròpia]

El disseny final no ha patit massa canvis en comparació amb el Mock-Up que es va fer, ja que com es pensava, només es va afegir text i una imatge d'un dish Stirling. Entre els canvis més destacats, es troba el fet d'incorporar un petit apartat on s'expliqués l'objectiu i la motivació que ha promogut la creació del dashboard, no només incloure informació del dish Stirling. A més, cal destacar que s'ha decidit canviar el títol d'aquesta pantalla per passar d'anomenar-se "Informació general" a anomenar-se "Inici". Aquest canvi va ser promogut perquè el fet de tenir títols amb accents provocava la presència d'errors en els codis de Node-Red creats per fer canvis estètics al dashboard.

A continuació es mostra una imatge del resultat final de la pantalla 1 del dashboard (Fig. 51). En aquesta imatge es pot veure tota la informació inclosa i com s'ha distribuït el text i la imatge. A la part dreta de la barra superior es veu el logo de l'ETSEIB i a la part esquerra de la mateixa barra apareix el títol de la pantalla i tres barres horitzontals que en prémer-les obren el menú lateral. Amb aquest menú es pot navegar entre les diferents pantalles del dashboard i el seu aspecte es pot veure a la captura 52.

Figura 51: Resultat final de la pantalla 1 del dashboard [Font pròpia]

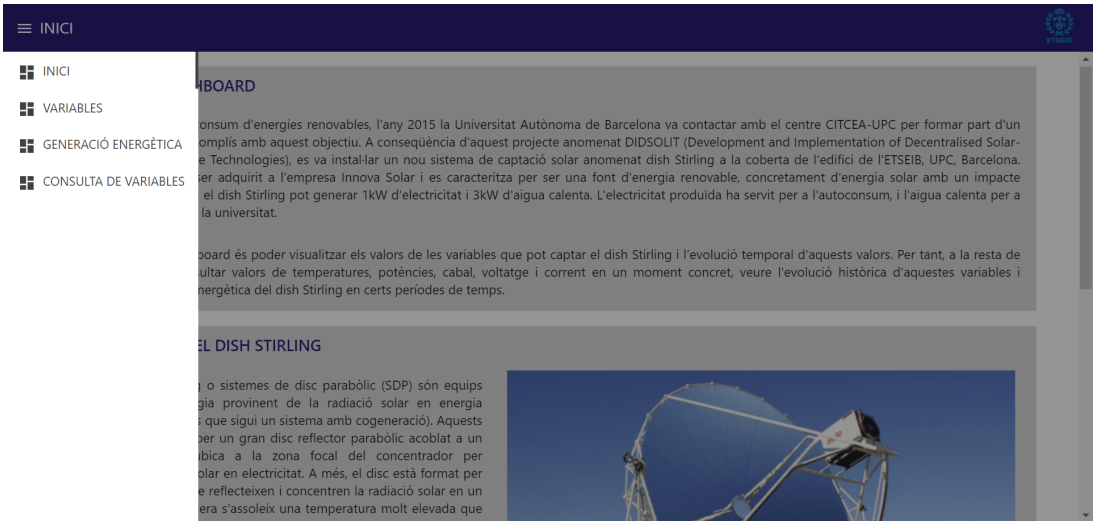


Figura 52: Menú lateral del dashboard [Font pròpia]

7.2 Pantalla 2. Variables

La segona pantalla del dashboard consisteix a veure l'evolució històrica de les variables del dish Stirling. L'elaboració d'aquesta pantalla mitjançant el software Node-Red va ser explicada a l'apartat 6.3.2. Aquesta pantalla incorporarà uns filtres per poder escollir el període i la variable de la qual es vol veure l'evolució històrica. Aquesta evolució es podrà observar mitjançant un gràfic. Seguidament es mostra una imatge del disseny inicial que es va fer d'aquesta pantalla:

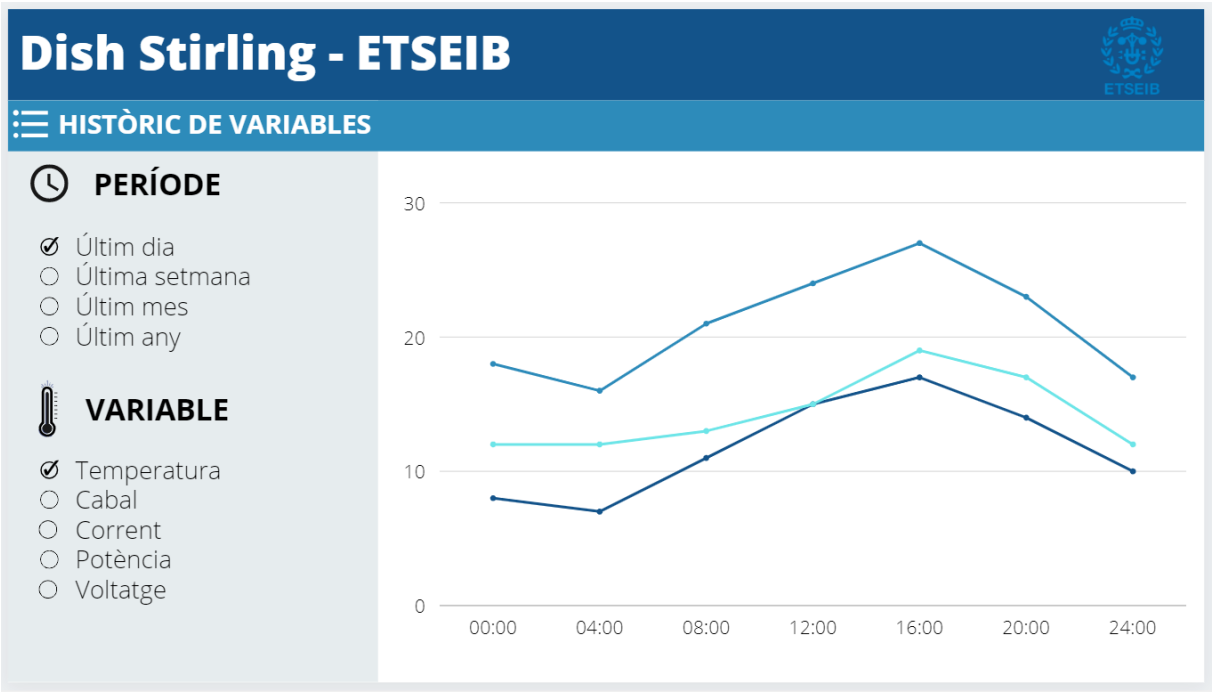


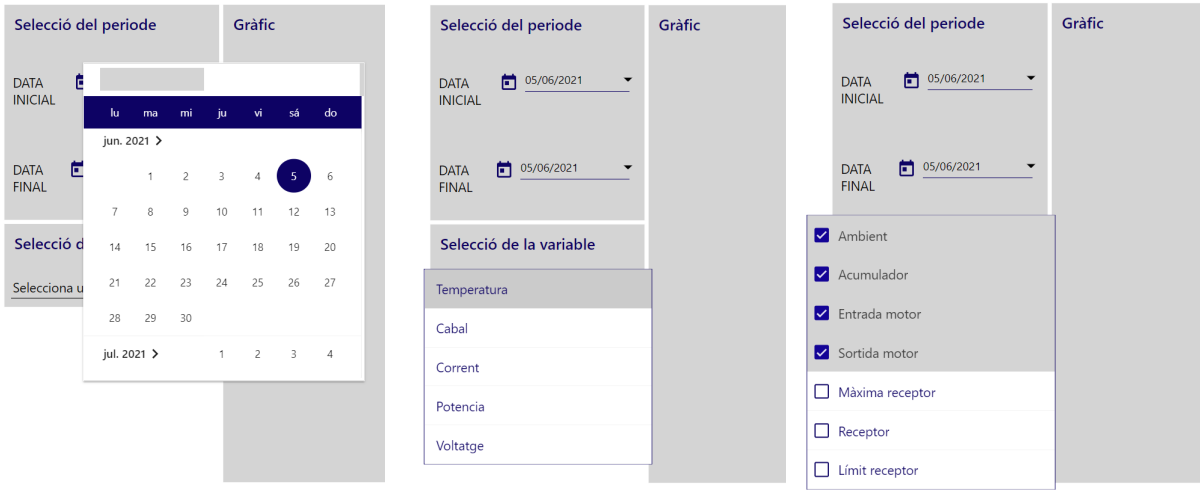
Figura 53: Mock-Up de la pantalla 2 del dashboard [Font pròpia]

Inicialment, la idea era que a la part dels filtres on s'indica el període, es pogués escollir entre algunes opcions com les que apareixen al Mock-Up (últim dia, última setmana, etc.). A mesura que es va anar desenvolupant el flow d'aquesta pantalla i es van adquirir més coneixements respecte al que es pot fer amb Node-Red, es va decidir que seria molt més interessant poder indicar el dia inicial i el dia final per definir el període. D'aquesta manera, no es tindria la limitació de només poder veure alguns períodes determinats. A més, quan es va fer el primer disseny de la pantalla, només es va pensar a fer un filtre per les variables i que, quan se seleccionés una variable que contingui més d'una variable (com per exemple la temperatura o la potència), al gràfic sortissin totes les temperatures o potències a l'hora. Més endavant, es va creure convenient que quan se seleccionés alguna d'aquestes dues variables aparegués un nou filtre per escollir quines de les temperatures o potències es vol que apareguin al gràfic (podent seleccionar més d'una). També es va canviar el títol de la pantalla que es va pensar inicialment ("Històric de variables") pel mateix motiu que el de la pantalla 1 i es va anomenar "Variables".

A continuació es mostra una imatge del resultat final de la pantalla 2 del dashboard (Fig. 54). Aquest és l'aspecte que té la pantalla el primer cop que s'entra. Al gràfic només apareix el missatge que es veu a la figura inferior (blank label) perquè encara no s'ha seleccionat cap variable. La data inicial i final mostra per defecte el dia actual.

Figura 54: Resultat final de la pantalla 2 del dashboard (1) [Font pròpia]

A la imatge 55a es pot veure l'aspecte del calendari quan es vol seleccionar el període. A la figura 55b es pot observar l'aspecte de la llista de les variables. En cas que se seleccioni a la llista de variables la temperatura o la potència, apareixerà un altre desplegable per escollir quines temperatures o potències es volen avaluar. A la captura 55c es veu a mode d'exemple la llista de temperatures que es poden seleccionar quan s'escull la variable temperatura. Cal destacar que a diferència de la llista de variables; a la llista de temperatures i a la llista de potències es poden seleccionar més d'una per poder veure-les a l'hora al gràfic.



(a) Selecció calendari (b) Selecció variable (c) Selecció temperatura

Figura 55: Aspecte dels filtres de la pantalla 2 del dashboard [Font pròpia]

Un cop seleccionades les temperatures cal prémer el botó "Acceptar" que es veu a la part inferior esquerra de la imatge 56 i d'aquesta manera apareixeran al gràfic les variables seleccionades. A la part superior del gràfic apareix la llegenda. Seleccionant les variables a la llegenda es poden amagar o fer aparèixer les variables que s'havien seleccionat. En el cas de la imatge, s'ha premut

la variable "Temperatura entrada motor" i, per tant, aquesta surt a la llegenda ratllada i no apareix al gràfic. A més, cal destacar que l'eix vertical del gràfic s'ajusta automàticament per mostrar les variables de la manera més adequada.

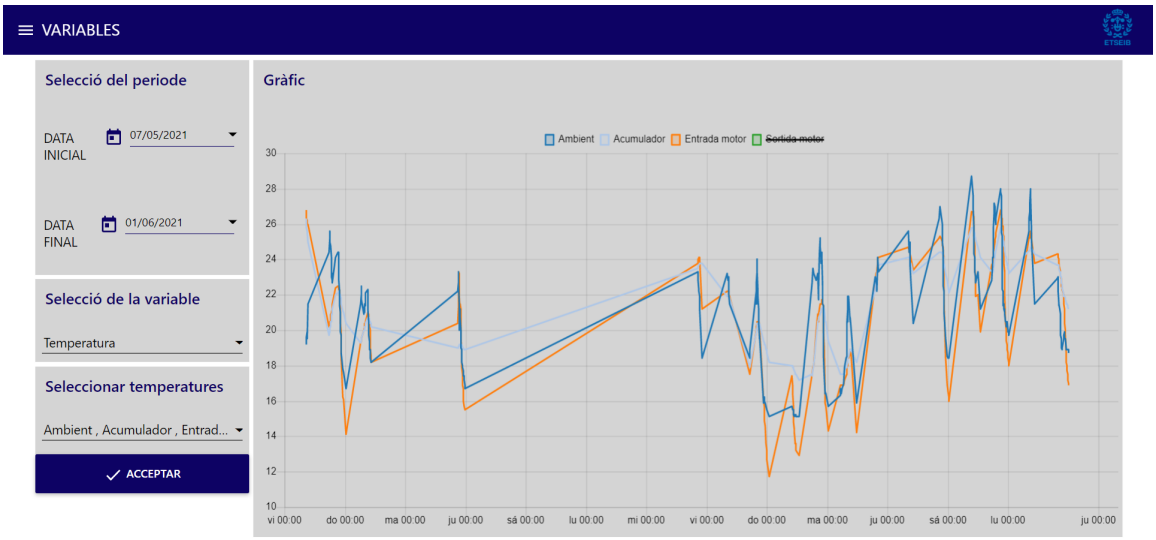


Figura 56: Resultat final de la pantalla 2 del dashboard (2) [Font pròpia]

7.3 Pantalla 3. Generació energètica

La tercera pantalla del dashboard consisteix a veure la generació d'energia elèctrica i tèrmica del dish Stirling. L'elaboració d'aquesta pantalla mitjançant el software Node-Red va ser explicada a l'apartat 6.3.3. Seguidament es mostra una imatge del disseny inicial que es va fer d'aquesta pantalla:

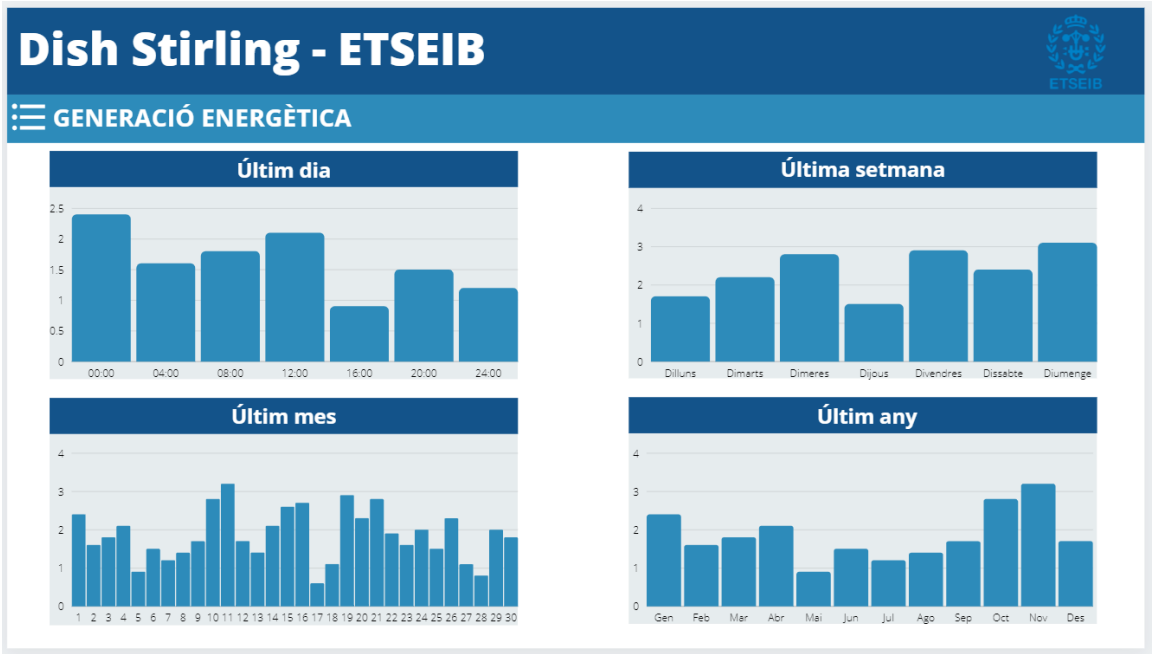


Figura 57: Mock-Up de la pantalla 3 del dashboard [Font pròpia]

La idea inicial d'aquesta pantalla era afegir 4 gràfics per veure la generació energètica en quatre períodes diferents (últim dia, última setmana, últim mes i últim any). Per tant, la primera idea de disseny consistia en una pantalla no interactiva, ja que només es podria visualitzar els gràfics. Una altra idea que va sorgir era la de mostrar només un gràfic i afegir un filtre per poder escollir entre aquests quatre períodes. Finalment, es va decidir mantenir els quatre gràfics tot i que en comptes de disposar-los perquè es veiessin els quatre a l'hora a la pantalla, s'han col·locat un sota l'altre, perquè d'aquesta manera els gràfics es mostren més amplis i es poden veure millor les dades. A més, s'han afegits botons per poder avançar o endarrerir el període que es visualitza al gràfic. Per tant, aquesta pantalla també és interactiva.

Figura 58: Resultat final de la pantalla 3 del dashboard [Font pròpia]

El resultat final d'aquesta pantalla es mostra a la figura 58. A la part superior s'ha afegit un filtre per poder escollir entre l'energia elèctrica i la tèrmica, element que tampoc es va considerar al disseny inicial. A més, s'han agregat polsadors a cada gràfic per poder avançar o retrocedir el període de temps que es visualitza. També apareix un text a sota de cada gràfic on es mostra la data inicial i final del període que s'està visualitzant i l'energia total generada en aquest interval de temps. Com s'observa a la imatge, a la major part dels períodes no hi ha cap energia generada. Això és degut al fet que durant la realització del projecte gran part del temps el dish Stirling no ha funcionat correctament i s'ha estat arreglant. A més, com es va explicar a l'apartat 6, només s'emmagatzemen les variables a la base de dades quan el Node-Red està encès i, per tant, no es tenen massa dades. Això se solucionarà instal·lant el programa que s'ha creat en un servidor Linux perquè pugui funcionar constantment.

7.4 Pantalla 4. Consulta de variables

La quarta i última pantalla del dashboard consisteix en la recerca dels valors de les variables del dish Stirling en un moment determinat. L'elaboració d'aquesta pantalla mitjançant el software Node-Red va ser explicada a l'apartat 6.3.4. Seguidament es mostra una imatge del disseny inicial que es va fer d'aquesta pantalla:

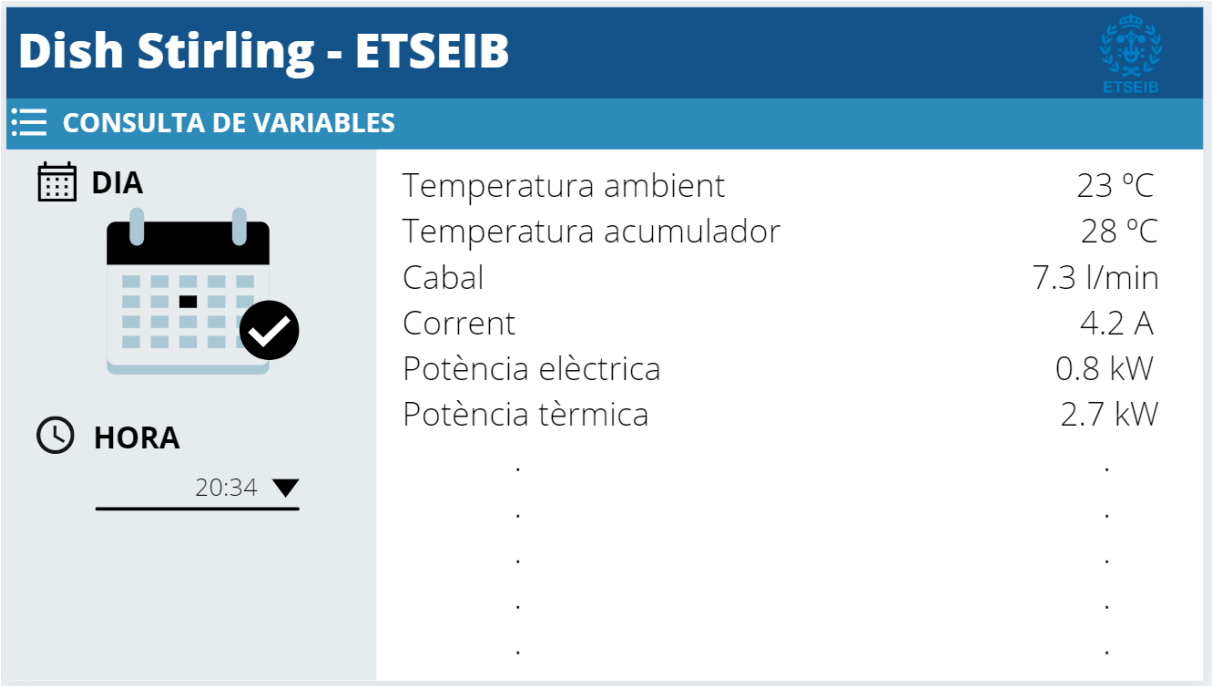


Figura 59: Mock-Up de la pantalla 4 del dashboard [Font pròpia]

Inicialment, la idea del primer disseny era fer uns filtres on es triés el dia i l'hora de l'instant que es vol consultar i un llistat de les variables amb el valor d'aquell moment. Per tant, és una pantalla interactiva. El resultat final no ha variat massa respecte el primer disseny tret de la distribució dels filtres i el text de les variables. A la imatge 60, es veu el resultat final de la pantalla 4 del dashboard. El primer cop que s'accedeix a aquesta pantalla del dashboard les variables no indiquen cap valor perquè no s'ha indicat un instant determinat.

Figura 60: Resultat final de la pantalla 4 del dashboard (1) [Font pròpia]

Quan es selecciona un instant determinat (dia i hora) apareix juntament amb el nom de cada variable el seu valor. S'han mostrat al centre les variables que fan referència a la potència generada (potències i potències sobre la nominal) perquè s'ha considerat que aquesta era la variable més rellevant. A la dreta destaca la variable: "Cabal de fluid refrigerant" que es mostra com un indicador de volum. El dibuix del dipòsit s'omple més o menys en funció de la quantitat de cabal (el mínim és 0 i el màxim s'ha determinat igual a 20 l/min).

Figura 61: Resultat final de la pantalla 4 del dashboard (2) [Font pròpia]

8 Planificació

En aquest apartat es definirà la planificació inicial del projecte, l'establerta durant les primeres setmanes del quadrimestre, la qual s'ha seguit fins a la finalització d'aquest. Per tant, s'explicaran les tasques a realitzar durant el projecte i la seva organització temporal. A més, s'inclourà un diagrama de Gannt per tal de poder advertir de manera visual la dependència temporal entre les tasques executades.

8.1 Calendari

S'ha tingut en compte com a data d'inici del projecte la data d'inici del quadrimestre, que ha sigut el dia 15 de febrer de 2021. Tot i això, l'elecció del tema del TFG es va realitzar abans i es van començar els treballs de documentació i aprenentatge la setmana abans de l'inici del quadrimestre. El treball s'haurà de dipositar entre els dies 14 i 21 de juny, data en la qual també l'haurà de validar el director del treball. Per tant, s'agafarà com a data de finalització del treball el 12 de juny per tenir marge fins a la data límit. D'aquesta manera, si hi hagués algun imprevist, es podria solucionar fins al dia 21. A més, s'ha de tenir en compte que hi ha una avaluació parcial entre els dies 17 i 21 d'abril. Per aquest motiu, s'haurà de fer el treball de manera progressiva i amb la intenció de tenir aproximadament la meitat del treball fet per l'avaluació parcial. També cal comentar que es durà a terme la defensa del treball entre els dies 5 i 16 de juliol, però no s'ha tingut en compte la preparació de la defensa en la planificació del projecte perquè aquesta es farà un cop s'hagi dipositat el treball.

Convocatòria Quadrimestre Primavera		
Matrícula ordinària	e-Secretaria [Automatrícula]	Dia assignat de la matrícula
Matrícula extraordinària per a l'estudiantat que no tingui registrat el treball en el període de matrícula ordinària	e-Secretaria [Instàncies -> altres]	Del 15 de febrer a l'1 de març de 2021
Avaluació parcial	Aplicació TFE	Del 17 al 21 d'abril de 2021
Dipòsit digital	Aplicació TFE	Del 14 al 21 de juny de 2021
Validació del director/ponent	Aplicació TFE	Del 15 al 22 de juny de 2021
Defenses	Aplicació TFE	Del 5 al 16 de juliol de 2021

Figura 62: Calendari del TFG per la convocatoria del quadrimestre de primavera [32]

Segons el pla d'estudis de la universitat, el TFG equival a 12 crèdits. Tenint en compte que cada crèdit implica 25 hores de treball, el temps dedicat al TFG hauria de ser d'aproximadament 300 hores. Tot i això, es considera que el treball podria necessitar més hores de treball i s'ha previst inicialment una durada de 350 hores. Aquest temps addicional s'ha inclòs perquè el treball consta de moltes tasques de programació, a les quals és difícil d'assignar una durada estimada amb precisió, ja que poden sorgir una gran varietat de problemes durant la realització de les

tasques d'aquesta tipologia. El treball per tant començarà a mitjans de febrer i acabarà a mitjans de juny, el que fa un total de 4 mesos o, més específicament, 17 setmanes. Es té previst dividir el total d'hores entre les 17 setmanes i, per tant, s'haurà de dedicar 20 hores i 35 minuts setmanals al TFG. Tot i fer aquesta divisió de manera exacta, en el moment d'haver de fer el treball, les hores dedicades setmanalment no seran igual durant cada setmana. Això es deu al fet que per exemple a la setmana santa es podrà dedicar més temps del previst i a la setmana d'exàmens de la universitat, es dedicaran menys hores.

8.2 Tasques del projecte

S'ha dividit el treball necessari per finalitzar correctament el projecte en diferents tasques. Dins de cadascuna d'aquestes tasques s'esmentaran les activitats que es duran a terme. A més, es definirà el temps dedicat a cadascuna de les tasques, les quals s'expliquen a continuació:

Planificació del projecte: Aquesta tasca consisteix en una primera presa de contacte amb el tema escollit pel projecte. En aquesta etapa inicial, es fa una primera anàlisi i disseny del projecte. Per tant, dins d'aquesta tasca es faran activitats relacionades amb una primera cerca d'informació per conèixer el tema que es tractarà. Un cop pensat i contextualitzat el tema del treball es realitza una planificació inicial amb les tasques que es realitzaran al llarg del projecte i definint un temps aproximat per a cadascuna d'aquestes tasques.

Definició dels objectius: La segona tasca consisteix en la definició dels objectius que es volen assolir amb la realització del treball i que es poden veure a l'apartat 1.3. La definició dels objectius ens permetrà tenir una idea més concreta del treball de la que es tenia després de fer la primera tasca (Planificació inicial). Dins d'aquesta activitat, també s'inclou una reunió amb el tutor, a la qual es va definir de manera conjunta la línia a seguir per fer el treball i es van aportar idees sobre quines eines podien ser utilitzades al llarg del projecte (Node-Red, MariaDB...).

Aprenentatge teòric de les eines utilitzades: Un cop definides les possibles eines a utilitzar, s'ha de fer una cerca d'informació per aprendre sobre aquestes eines escollides. En el cas del present projecte, aquesta cerca de documentació i aprenentatge estarà relacionat amb les eines utilitzades que requereixen uns mínims de coneixement: JavaScript, Node-Red i MariaDB.

Implementació: La quarta tasca és a la que es dedicarà més temps. Consisteix en l'aplicació de tot el coneixement adquirit en la tasca anterior i en l'assoliment dels objectius del treball definits a la segona tasca. Dins d'aquesta tasca trobem diverses activitats:

- Creació de la base de dades.
- Introducció de les variables a la base de dades.
- Creació i disseny inicial del Dashboard.
- Millora del disseny del Dashboard.

Anàlisi de les dades i els resultats obtinguts: Consisteix a extreure conclusions a partir de l'anàlisi dels resultats mostrats als gràfics del Dashboard.

Redacció de la memòria: Consisteix en la redacció de l'informe final on s'explicarà tot el procés seguit durant la realització del projecte i es finalitzarà amb les conclusions obtingudes de tot aquest treball.

8.3 Temps estimat i distribució temporal del projecte

A continuació es mostra una taula (Taula 5) amb el temps dedicat a cadascuna de les tasques. A més, s'han dividit les tasques en les activitats que les componen per poder estimar la dedicació temporal amb una precisió major. Com es pot observar, la implementació és el bloc al qual es dedicarà més temps de treball, ja que el present projecte és un treball principalment pràctic. També es dedicarà molt de temps a l'aprenentatge necessari per realitzar la implementació i a l'explicació del treball realitzat mitjançant la redacció de la memòria.

TASQUES DEL PROJECTE	DEDICACIÓ (en hores)
Planificació del projecte	**13 hores**
Primera cerca d'informació	5
Contextualització del projecte	2
Planificació temporal inicial	4
Instal·lació dels softwares necessaris	2
Definició dels objectius	**4 hores**
Reunió amb el tutor del projecte	1
Definició dels objectius i de l'abast del projecte	3
Aprenentatge teòric de les eines utilitzades	**80 hores**
Aprenentatge sobre MariaDB	12
Aprenentatge sobre JavaScript	18
Aprenentatge sobre Node-Red	50
Implementació	**175 hores**
Creació de la base de dades	2
Introducció de les variables a la base de dades	38
Creació i disseny inicial del Dashboard	110
Millora del disseny del Dashboard	25
Anàlisi de les dades i els resultats obtinguts	**8 hores**
Anàlisi dels resultats	4
Extracció de conclusions	4
Redacció de la memòria	**70 hores**
Redacció de la memòria	60
Revisió de la memòria	10

Taula 5: Taula amb la dedicació en hores de les diferents tasques [Font pròpia]

A més d'incloure una taula amb la dedicació temporal, s'ha estimat oportú mostrar la dependència temporal entre aquestes tasques. Per fer-ho, s'ha afegit el Diagrama de Gannt que es mostra a continuació (Fig. 63). El diagrama s'ha elaborat utilitzant el software d'administració de projectes anomenat Microsoft Project. A la part esquerra de la imatge es veuen les tasques i les activitats fetes i a la part superior es veu l'escala temporal durant la qual s'ha dut a terme el projecte. La línia vertical verda marca la data límit del projecte (21 de juny de 2021) tot i que, com s'ha comentat i, tal com es pot veure al diagrama, es preveu que el projecte estigui finalitzat el dissabte 12 de juny de 2021. A la part dreta de la imatge es veuen les barres que corresponen a cadascuna de les activitats, i que comprenen el període de temps en el qual es realitzaran aquestes. Finalment, es mostren fletxes blaves que uneixen alguna de les barres per indicar la dependència temporal entre alguna de les activitats.

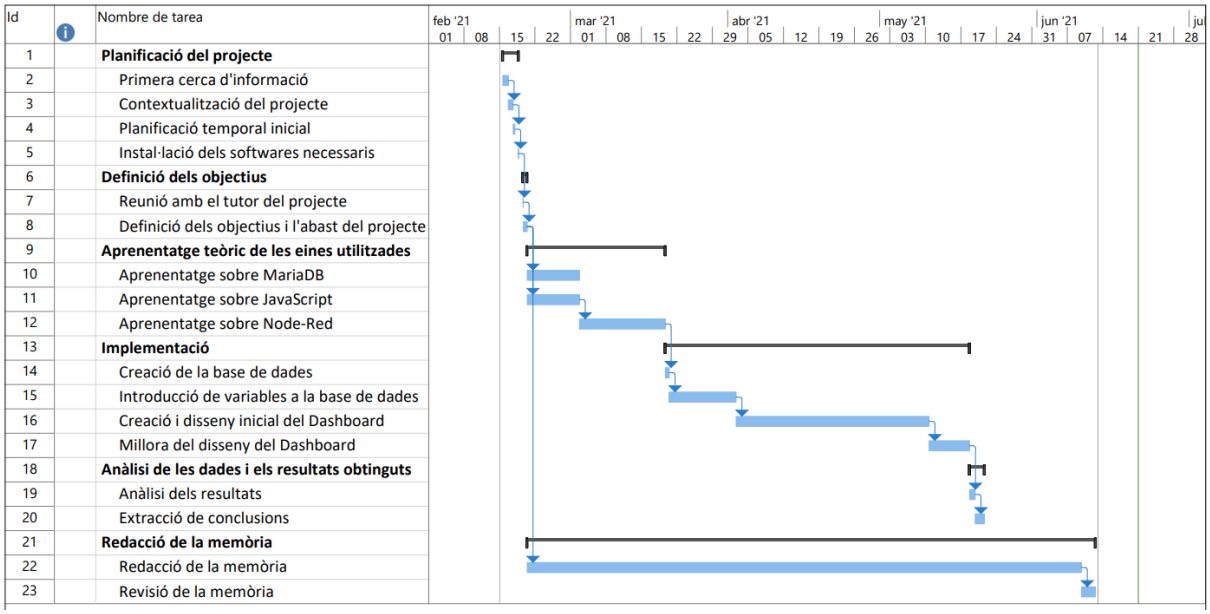


Figura 63: Diagrama de Gannt del projecte [Font Pròpia]

9 Pressupost

A continuació, s'elaborarà un pressupost aproximat per avaluar el cost de l'elaboració del projecte tenint en compte els recursos emprats i la mà d'obra.

9.1 Cost de personal

En primer lloc, per realitzar l'estudi econòmic s'han analitzat els costos de personal. En aquest apartat, s'ha de considerat el cost degut al sou pel treball realitzat. Les tasques que s'han dut a terme durant el projecte haurien de ser realitzades per tres tipus de treballadors diferents:

- **Enginyer:** Encarregat de la planificació, d'analitzar i extreure conclusions de resultats i de donar idees per a la implementació del projecte. Les tasques del projecte en les quals està implicat són: Planificació del projecte, definició dels objectius, part de la implementació (una tercera part), anàlisi de les dades i els resultats obtinguts i revisió de la memòria. Es considera que el seu treball implica un total de 95 hores i percebrà un sou de 40 €/hora.

- **Programador:** Encarregat de la part de programació del projecte, basant-se en les decisions que hagi pres l'enginyer en aspectes com per exemple com realitzar el disseny del Dashboard. Les tasques del projecte en les quals està implicat són: Aprenentatge teòric de les eines utilitzades i Implementació (dues terceres parts). Es considera que el seu treball implica un total de 195 hores i percebrà un sou de 30 €/hora.

- **Personal Administratiu:** Encarregat de l'aspecte formal del projecte i de plasmar el treball de la resta de treballadors a la memòria. Les tasques del projecte en les quals està implicat són: Redacció de la memòria. Es considera que el seu treball implica un total de 60 hores i percebrà un sou de 20 €/hora.

A la taula que es mostra a continuació (Taula 6) es mostra el cost de personal desglossat per cada tipus de treballador implicat en el projecte. El cost de personal total del treball serà de 10.850 €.

TREBALLADOR	COST (€/hora)	TEMPS DE TREBALL (hores)	COST TOTAL (€)
Enginyer	40	95	3.800
Programador	30	195	5.850
Personal administratiu	20	60	1.200
TOTAL			10.850

Taula 6: Taula amb el cost degut a les hores de cada treballador [Font pròpia]

A continuació, s'ha inclòs també una taula on es pot veure el cost de personal per cadascuna de les tasques del projecte (Taula 7). A la taula, s'ha afegit els treballadors implicats a cada tasca i quant temps ha dedicat el treballador a cada tasca concreta. Després, s'ha multiplicat el cost per hora del treballador pel nombre d'hores que realitza per obtenir el cost de cada tasca. Com era previsible, les tasques amb més pes econòmic són l'*Aprenentatge teòric de les eines utilitzades* i la *Implementació*.

TASCA DEL PROJECTE	TREBALLADORS IMPLICATS	TEMPS (hores)	COST (€)
Planificació del projecte	Enginyer	13	520
Definició dels objectius	Enginyer	4	160
Aprenentatge teòric de les eines utilitzades	Programador	80	2400
Implementació	Programador Enginyer	115 60	3450 2400
Anàlisi de les dades i els resultats obtinguts	Enginyer	8	320
Redacció de la memòria	Personal administratiu Enginyer	60 10	1200 400
TOTAL			10.850

Taula 7: Taula amb el cost degut a les hores dedicades a cada tasca del projecte [Font pròpia]

9.2 Costos informàtics i d'equipament

En aquesta secció s'estudiaran els costos dels equipaments utilitzats i els costos informàtics derivats del desenvolupament del projecte. Dins d'aquests últims costos s'inclou tant els recursos de hardware com els de software.

Per calcular el cost de l'ordinador es calcularà el cost d'amortització d'aquest durant les hores que s'ha utilitzat per al projecte. Per determinar el cost d'amortització s'ha tingut en compte que la vida útil d'un ordinador portàtil és de 5 anys. Per a cada any s'han establert 1780 hores de treball. Per tant, per calcular el cost d'amortització es farà la següent fórmula:

$$\text{Cost d'amortització } [€/h] = \text{Preu de l'ordinador } [€] \ / \ (5 \text{ anys} \cdot 1780 \text{ hores/any})$$

D'aquesta fórmula s'obté un cost d'amortització de 0,101 €/hora. L'ordinador utilitzat pel projecte és un portàtil de la marca *hp* amb un processador *Intel Core i7 8th Gen* que té un cost de 900 €. Per calcular el temps d'utilització de l'ordinador portàtil s'ha tingut en compte que l'ordinador s'ha utilitzat per executar totes les tasques del projecte amb l'excepció de part de l'aprenentatge teòric (15 hores). D'aquesta manera quedaria un total de 335 hores d'ús de l'ordinador. Donat que perquè s'afegissin variables a la base de dades l'ordinador havia d'estar encès amb el Node-Red executat, en moltes ocasions es va deixar l'ordinador engegat per poder tenir més dades i poder fer els gràfics amb més precisió. Per tant, al temps d'ús de l'ordinador s'han d'afegir 50 hores addicionals dedicades a mantenir el Flow de Node-Red executat, tot i que en aquests moments no s'estava fent cap tasca del treball. Llavors quedaria un ús total de l'ordinador de 375 hores.

Per la part del software, s'ha tingut en compte que el cost d'una llicència de Microsoft Office és de 69 € per un any. La resta de softwares utilitzats són softwares lliures que no tenen cap cost. Per calcular l'amortització del paquet office s'utilitzarà una fórmula similar a la de l'amortització del portàtil:

$$\text{Cost d'amortització } [€/h] = \text{Preu de la llicència } [€] \ / \ (1 \text{ any} \cdot 1780 \text{ hores/any})$$

D'aquesta fórmula s'obté un cost d'amortització de 0,039 €/hora. El temps en què s'han utilitzat els programes de Microsoft Office ha sigut el temps emprat en la realització de les tasques de *Redacció de la memòria* i *Planificació temporal inicial*. Per tant, s'ha fet un ús dels programes inclosos en la llicència de Microsoft Office de 74 hores.

Finalment, per la part dels costos d'equipament s'ha de tenir en compte el llibre *Domine JavaScript. 4a edición* de Pablo E. Fernández Casado [27] que es va utilitzar per a l'aprenentatge del llenguatge JavaScript i que té un cost de 28,41 €.

EQUIPAMENT	COST (€)	UNITATS	AMORTITZACIÓ (€/h)	Ús (hores)	COST TOTAL (€)
Ordinador portàtil hp	900	1	0,101	375	37,875
Llicència de Microsoft Office 2013	69	1	0,039	74	2,886
Llibre Domine JavaScript	28,41	1	-	-	28,41
TOTAL					69,171

Taula 8: Taula amb el costos informàtics i d'equipament [Font pròpia]

Per calcular els costos informàtics i d'equipament que es veuen a la taula superior (Taula 8), s'ha multiplicat l'amortització per les hores d'ús per obtenir el cost total. En el cas del llibre, s'ha agafat directament el cost del producte. Finalment, s'obté que els costos informàtics i d'equipament totals són de 69,17 €.

9.3 Altres costos

En aquest apartat s'estudiarà el cost energètic del projecte i el cost per la connexió a Internet. Cal destacar, que el fet de fer aquest treball durant la pandèmia de la COVID-19 ha fet que no existissin costos per transport, ja que s'ha reduït l'activitat a la universitat i, per tant, les reunions i comunicacions amb el tutor del projecte s'ha fet telemàticament.

Pel que fa als costos energètics, s'ha tingut en compte el cost d'energia per carregar l'ordinador. El carregador de l'ordinador consumeix una potència de 0,065 kW i triga aproximadament 4 hores a realitzar una càrrega completa. A més, s'ha considerat un preu per al consum elèctric de 0,14 €/kW. Per tant, el cost de fer una càrrega complerta de l'ordinador portàtil és de 0,036 €. S'ha considerat que l'autonomia de la bateria del portàtil és de 4 hores i, com que durant el treball s'ha utilitzat l'ordinador durant 375 hores, s'han fet 94 càrregues completes de la bateria. Això completa un cost energètic total de 3,42 €.

L'altre cost que s'ha tingut en compte és el cost per l'Internet utilitzat. El cost mensual per la fibra òptica és de 40 €. El treball s'ha realitzat en un període de 4 mesos i, per tant, s'han de pagar 160 € d'Internet.

A la taula inferior (Taula 9), es pot veure un resum de la informació detallada anteriorment, i s'indica que el cost total d'aquests costos addicionals és de 163,42 €.

CONCEPTE	PREU	QUANTITAT	COST TOTAL (€)
Cost energètic	0,14 €/kW	94 càrregues de 4 hores	3,42
Connexió a Internet	40 €/mes	4 mesos	160
TOTAL			163,42

Taula 9: Taula amb el cost energètic i de connexió a Internet [Font pròpia]

9.4 Cost total

Amb la suma de tots els costos estudiats als apartats anteriors trobarem el cost total del projecte. Tot i això, a aquest preu se li afegirà un 10 % per possibles imprevistos (com per exemple una avaria de l'ordinador, haver de fer més hores de les calculades, etc.) i per calcular alguns costos indirectes no avaluats, com per exemple el cost de la llum. A la taula següent (Taula 10), es veuen tots aquests costos explicats, i la suma total del pressupost, que és de 12190,85 €. Com es pot observar a la taula, el cost principal és el cost de personal, que representa un 89 % de tot el pressupost.

COST	PREU (€)
Cost de personal	10.850
Costos informàtics i d'equipament	69,171
Cost energètic i connexió a Internet	163,42
TOTAL	11082,59
Costos indirectes i imprevistos (10%)	1108,26
COST TOTAL	**12190,85**

Taula 10: Taula amb el cost total del projecte [Font pròpia]

10 Impacte ambiental

Com es va explicar a la introducció del treball (1), cada any s'està augmentat el consum d'energies renovables per intentar pal·liar els efectes negatius de la contaminació produïda per l'ús de fonts d'energia no renovables. Per tant, sempre que es du a terme l'execució d'un projecte és imprescindible avaluar quin impacte ambiental pot tenir aquest per tal d'evitar o reduir l'impacte negatiu que el projecte té al medi ambient. En aquest sentit, a l'apartat actual s'estudiarà l'impacte ambiental del present treball.

L'impacte ambiental negatiu d'aquest projecte és mínim perquè s'ha emprat una quantitat molt reduïda de recursos que poden resultar perjudicials per al medi ambient. Concretament, s'analitzarà el consum elèctric produït per la realització del treball com l'única variable causant d'un impacte mediambiental negatiu. Com es va comentar a l'apartat del pressupost (9.3), a causa del treball es va carregar la bateria d'un ordinador portàtil 94 cops durant 4 hores. La potència d'aquest carregador és de 0,065 kW, per tant el consum energètic total és de 24,44 kWh. S'ha tingut en compte que el factor de conversió per passar de kWh a emissió en grams de CO2 és de 0,357 kg CO2/kWh [33]. Multiplicant els kWh consumits durant tot el treball per aquest factor obtenim que la quantitat total de kg de CO2 emesos a causa de la realització del treball és de 8,73 kg CO2. Aquesta quantitat no és massa gran si la comparem amb les principals fonts d'emissió de CO2 de l'actualitat, com poden ser els cotxes o les fàbriques. Per tant, l'impacte ambiental del treball és relativament baix. Tot i la producció de gasos CO2 a causa d'aquest projecte, es considera que aquest impacte ambiental negatiu serà molt poc significatiu en comparació amb l'impacte positiu que pot produir. El desenvolupament d'un bon sistema de monitoratge i de visualització pot millorar significativament el rendiment d'un sistema de captació solar, ja que permet detectar amb facilitat possibles problemes del sistema o advertir si el sistema no està funcionant correctament. Això és encara més important si tenim en compte que el sistema dish Stirling instal·lat a la coberta de l'edifici de l'ETSEIB i estudiat al present treball ha tingut nombroses fallides durant els últims anys.

Com s'ha explicat durant el treball, el sistema dish Stirling produeix energia elèctrica mitjançant la captació de radiació solar. Aquest mètode de producció d'electricitat té una repercussió mínima al medi ambient en comparació amb la contaminació emesa en la producció d'electricitat amb energies primàries no renovables. Millorar el rendiment del dish Stirling faria augmentar la quantitat d'electricitat emprada a la universitat que prové d'una energia renovable, ja que l'energia produïda s'utilitza per escalfar l'aigua del gimnàs. Per tant, això seria beneficiós pel planeta, gràcies al fet que es reduiria la contaminació en la producció d'electricitat amb fonts no renovables com: el petroli, el carbó, el gas natural, etc. Amb un funcionament correcte, el sistema dish Stirling estudiat produeix una energia elèctrica mitjana anual de 2100 kWh i una energia tèrmica mitjana anual de 6400 kWh amb una capacitat d'escalfar 80 litres d'aigua per hora. Amb l'anàlisi feta anteriorment, s'ha determinat que s'han consumit 24,44 kWh per fer el treball, el qual representa un 1,16 % de l'energia elèctrica que pot produir el dish Stirling en un any. Per tant, amb una mínima millora del sistema dish Stirling estudiat en aquest projecte gràcies al servidor web creat, ja es compensaria l'impacte mediambiental negatiu causat pel consum elèctric.

En conclusió, es pot determinar que l'impacte ambiental d'aquest projecte és pràcticament insignificant, ja que s'han emprat molt pocs recursos. Inclús, es pot considerar que l'impacte ambiental del treball és positiu per tota l'electricitat procedent de fons renovables que pot ajudar a generar després de la implementació del sistema de monitoratge.

Conclusions

Per concloure el treball s'hauran d'avaluar els objectius definits inicialment (apartat 1.3) per indicar si aquests s'han complit de manera total o parcial o si no s'han pogut assolir. Els tres primers objectius consistien en l'estudi i investigació del sistema dish Stirling. Concretament, l'objectiu era analitzar el funcionament de la tecnologia Stirling, estudiar el sistema dish Stirling de l'ETSEIB i entendre com aquest dish Stirling realitzava la recollida de dades. Es considera que aquests objectius s'han complert totalment durant les primeres setmanes del treball. Es considera que la cerca d'informació realitzada ha sigut suficient per adquirir els coneixements necessaris per poder dur a terme el projecte i assolir més endavant els següents objectius.

El quart objectiu era l'objectiu principal del projecte i consistia en la realització del dashboard utilitzat per visualitzar i monitorar les dades aportades pel dish Stirling. Es considera que s'ha complit aquest objectiu totalment i que s'ha aconseguit fer un dashboard que compleix amb els requeriments previs i que té les funcionalitats necessàries. L'últim objectiu consistia a analitzar les dades i resultats obtinguts per tal de garantir el correcte funcionament del dish Stirling i poder calcular valors com la generació energètica en certs períodes de temps o el rendiment del disc parabòlic de l'ETSEIB. Aquest objectiu no s'ha pogut assolir correctament degut a problemes durant l'elaboració del treball. Com s'ha comentat al llarg de la present memòria, el Node-Red només introduïa els valors de les variables a la base de dades quan estava el software encès. Per tant, l'emmagatzematge de les dades es fa de manera irregular i no periòdicament de manera exacta. Això impedeix que es pugui fer una anàlisi del funcionament del dish Stirling de manera correcta, ja que per exemple no es tenen pràcticament dades en hores nocturnes i, pot ser, hi ha períodes de temps llarg en què no hi ha cap dada. Un altre problema que ha impedit l'assoliment d'aquest objectiu de manera encara més greu, ha sigut que durant el temps d'execució del present treball, el funcionament del dish Stirling ha sigut molt irregular. Durant pràcticament tot el període en què s'ha fet el projecte, el dish Stirling ha estat espatllat i, per tant, la generació energètica en pràcticament tots els dies és de 0 kWh. Tot i això, sí que s'han pogut adquirir valors d'algunes variables com per exemple les temperatures perquè els sensors que mesuraven aquestes variables continuaven funcionant.

Per tant, es considera que s'han complit la majoria dels objectius en la mesura del possible. Tot i que els problemes esmentats han impedit que es pogués assolir també l'últim dels objectius del treball, es considera que la realització d'aquest ha sigut satisfactòria. S'espera que amb aquest treball es pugui millorar el control del funcionament del dish Stirling per tal d'aprofitar al màxim a l'edifici de l'ETSEIB l'energia que aquest pot generar.

Futures línies de treball

Com en qualsevol projecte, el present treball ha tingut una limitació temporal que ha impedit poder haver dut a terme tasques més enllà de les realitzades. En aquest sentit, es considera interessant comentar quines poden ser aquestes futures línies de treball.

Com que la tasca principal del projecte ha sigut la realització del dashboard, es podrien fer múltiples millores d'aquest. Es podrien fer canvis estètics o afegir noves funcionalitats al dashboard. Per exemple, donats els problemes de funcionament que està tenint el dish Stirling, es considera interessant poder afegir la possibilitat que s'avisi al tutor del treball quan el dish Stirling deixa de funcionar, enviant un correu electrònic per exemple. També es podria afegir alguna manera diferent per visualitzar les variables del dish Stirling, tot i que s'ha intentat fer el dashboard per-

què es pugui consultar qualsevol valor amb la màxima senzillesa i de la manera més entenedora possible.

La futura línia de treball més immediata i necessària és la instal·lació del flow de Node-Red al servidor Linux del tutor del treball, ja que com s'ha comentat anteriorment el Node-Red només introdueix els valors de les variables a la base de dades quan estava el flow engegat. Per tant, s'haurà d'instal·lar el software al servidor per tal que s'introdueixin les dades constantment i es pugui avaluar millor el funcionament del dish Stirling. També, això servirà per poder accedir al dashboard de manera externa, no només des del servidor local.

Fent aquests treballs s'aconseguiria millorar la funcionalitat del dashboard i es podria avaluar el correcte funcionament del dish Stirling. A més, un cop es pugui fer un anàlisi del rendiment del dish Stirling, es podria estudiar com millorar aquest rendiment per tal de que augmenti a l'ETSEIB el consum d'energia provinent de fonts renovables.

Agraïments

En primer lloc, voldria agrair al tutor d'aquest Treball Fi de Grau, Daniel Montesinos, el seu suport i orientació en la realització d'aquest projecte. El seu suport i els seus consells han resultat claus per poder assolir el projecte satisfactòriament. A més, voldria agrair-li que em donés l'oportunitat de fer aquest treball.

Finalment voldria agrair també a la meva família i a totes aquelles persones que tot i no participar directament al treball, el seu ajut i suport ha sigut imprescindible durant la realització del projecte.

Bibliografia

[1] APPA Renovables, *Renovables en el mundo y en Europa*, 3 de març de 2021

[2] Grups de recerca UAB, *Projecte DIDSOLIT*, 3 de març de 2021

[3] Campus Energia UPC, *Nou sistema de captació solar Dish Stirling instal·lat a l'ETSEIB*, 3 de març de 2021

[4] Wikipedia, *Motor Stirling*, 5 de març de 2021

[5] Demotor, *Història del motor Stirling*, 5 de març de 2021

[6] Madrid+D, *Una nova oportunitat. El disc Stirling*, 5 de març de 2021

[7] Energías Renovables, *S'aprova un projecte solar termoelèctric amb discos parabòlics de 300 MW a Califòrnia*, 5 de març de 2021

[8] LACYQS, *Discs parabòlics*, 6 de març de 2021

[9] A. Hafez, *Solar parabolic dish Stirling engine system design, simulation and thermal analysis*, ScienceDirect, 2016

[10] ENERGY.GOV, *Project Profile: Low-Cost, Lightweight Solar Concentrators*, 8 de març de 2021

[11] ANTENAS LEÓN, *CÁLCULO DE LA ELEVACIÓN, AZIMUT Y POLARIZACIÓN*, 8 de març de 2021

[12] BIBING, *INTRODUCCIÓ AL MOTOR STIRLING*, 10 de març de 2021

[13] Solar Energía, *MOTOR STIRLING Y CICLO STIRLING*, 10 de març de 2021

[14] BIBING, *Estudio Económico y Termoeconómico del Disco Solar Stirling EuroDish*, 11 de març de 2021

[15] GUIADEV, *MySQL vs SQLite*, 22 de març de 2021

[16] HOSTINGER, *SQLite vs MySQL: ¿Cuál es la diferencia?*, 22 de març de 2021

[17] CIRCUTOR, *Guia de funcionament del Gestor d'Eficiència Energètica EDS*, 1 d'abril de 2021

[18] UPCOMMONS, *Estudi d'un generador Stirling per aigua calenta i electricitat*, J. Marías, 1 d'abril de 2021

[19] MARIADB, *Informació sobre el sistema de gestió de base de dades MariaDB*, 4 d'abril de 2021

[20] STYDE, *Què és i per a què serveix SQL?*, 5 d'abril de 2021

[21] BAEHOST, *Llistat de comandes SQL bàsiques per MariaDB*, 7 d'abril de 2021

[22] MARIADB, *Informació sobre les comandes SQL bàsiques per MariaDB*, 7 d'abril de 2021

[23] APRENDIENDO ARDUINO, *Què és Node-Red?*, 11 d'abril de 2021

[24] NODE-RED, *Node-Red. Low-code programming for event-driven applications*, 11 d'abril de 2021

[25] MDN WEB DOCS, *Fundamentos de JavaScript*, 12 d'abril de 2021

[26] Javier Eguíluz Pérez, *Introducción a JavaScript*, 2009

[27] Pablo E. Fernández Casado, *Domine JavaScript. 4a edición*, 2020

[28] OPENWEBINARS, *Qué es Node.js y para que sirve*, 12 d'abril de 2021

[29] NEXTU, *JSON: ¿Qué es y para qué sirve?*, 13 d'abril de 2021

[30] MDN WEB DOCS, *Trabajando con JSON*, 13 d'abril de 2021

[31] HOSTINGER, *¿Qué es JSON?*, 13 d'abril de 2021

[32] ETSEIB - UPC, *Calendari tràmits acadèmics estudis de Grau*, 24 d'abril de 2021

[33] Energia.gob, *Calendari tràmits acadèmics estudis de Grau*, 30 d'abril de 2021