# Välkommen till tema-uppgift kring uppkopplad båt!

Det här dokumentet innehåller lite bakgrund och problem(möjlighet!)beskrivning kring uppkopplad båt. Vi kommer att ha expertis på plats som kan svara på frågor.

# En väska med teknik, en router och en Raspberry.

Framför er på bordet så finns ett utbildningskit från Garmin / Empirbus. Den innehåller alla kanaler som förekommer på en båt. Utrustningen har konfigurerats för att innehålla några olika typer av kontroller och värden som går att följa och manipulera via en webbläsare genom att surfa till den s.k WDU (Web display unit) som finns i systemet.

Setup:en där en Raspberry är ansluten till ett nät som har koppling till Garmin-utrustningens nät är identisk med hur det ska fungera på en båt.

## The basics

**Uppkoppling** på nätet för routern är Garmin/ Empirbus (nedan kallad båten) via nätverksingång samt Raspberry PI.

Båten är ansluten till en router som i sin tur har uppkoppling till Internet (i det här fallet via DHCP på H4:as fasta nät). Ni i teamet kan ansluta till den här routern för att komma åt Raspberry resp den interaktiva html-sidan som genereras från WDU:n.

### Anslut till routern - via wifi el trådat

NIC: hackathon\_1 (el 2)  
PWD: hackaddthon  
  
**Viktigt** – Garmin-utrustningen har en ”aggressiv”/ej påverkbar DHCP-inställning, så klienter som ansluter måste ställa om till statisk IP, ändra subnet-mask, peka ut en gateway samt dns-server manuellt för att det skall fungera. Så här ser det ut t.ex. i windows 11

Graphical user interface, text, application, email

Description automatically generated

* Routern (tillika gateway) har fast ip för anslutna:   
  172.16.67.1
* Subnet mask ska vara 255.255.0.0
* Din IP sätts till ett nummer som inte krockar med övriga team-medlemmar i spannet:

172.16.16.60 - 172.16.16.69. (En varning om du har Mac – när det här värdet ändras till statisk ip så gäller det globalt, byter du till annat wifi så slå av statiskt ip)

Routern   
För att logga in på routens admin-sida (vilket inte ska behövas egentligen, men ut i fall):  
U: admin  
P: hackaddthon

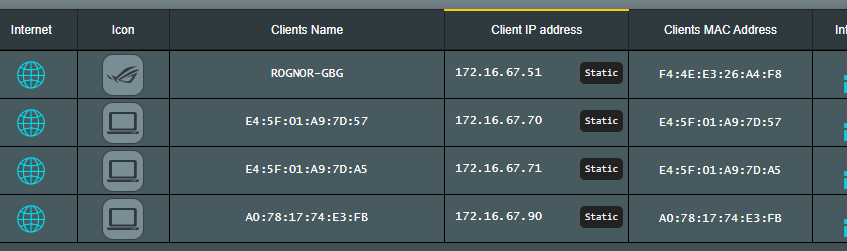
### WDU

IP-nummer till WDU – och webbservern – är inte helt förutsägbart. Genom att nyttja manöver-displayen i väskan (home > vessel > settings ) så går det att få fram vilket ip-nummer den har.   
Graphical user interface

Description automatically generated

På WDU:ns IP-adress finns en webbserver som returnerar de applikationer som konfigurerats och laddats upp på WDU:n (via USB).

Raspberry PI kommer att ha en fast IP-adress uppkopplad på sin router: 172.16.67.70|71 (ert team har en av dessa)



## Förutsättningar

WDU tillhandahåller en webbsida som fungerar som en applikation. Enda sättet att nå den är att vara ansluten till Garmin/Empirbus nät.   
Exempel på hur sidan kan se ut:  
  
Graphical user interface, application

Description automatically generated

Den här sidan innehåller ett enkelt värde som representerar en sensor i båten (”Tank level”).   
Den innehåller en alarm-nivå kopplad till värdet i Tank Level som är justerbar m.h.a. knapparna liksom slider-kontrollen. Om värdet understiger alarmnivån triggas ett larm, där ljud på en liten buzzer dyker upp, samt en lampa som blinkar ”rött”. I det här läget kan man endera ”tanka” genom att vrida om potentiometer (liten blank rund ”skruv” som är ansluten) eller ändra alarmnivån. För att bekräfta att larmet är noterat och återställa får man klicka på symbolen för varning (som visas istället för triangeln i en ring). När larmet inte längre är aktivt blinkar en lampa ”grönt” för att bekräfta det.

Sidan håller sig i synk med hårdvaran genom socket-anrop. När sidan laddas saknas värden, vilka fylls på mha socket-anrop. Om man inspekterar DOM:en kan man se att de olika div:arna i sidan har ett data-attribut som också råkar vara vilken kanal som representerar funktionen (data-ebp-component-id).

Inspekterar man trafiken som går via sockets så återfinns dessa id:n kopplat till vad som sker i systemet eller i UI.

### WDU kan ha flera sidor..

I den här setup:en du ser så ligger applikationen (html-sidan) som ”index.html” på WDU. Det kommer att finnas fler sidor (applikationer/konfigurationer) på en WDU. Den första som laddas upp får namnet index.html. Nästa applikation får namnet index1.html, index2.html osv. Den sida som är aktuell för den här integrationen vi har kommer att innehålla texten ”apiSSRS”.

# Bakgrund och uppgift (tema)

SSRS har mängder av sensorer och uppkoppling till sina båtar. De har också 2400 frivilliga sjöräddare med varierande bakgrund och erfarenhet. Det finns ett önskemål om att kunna läsa av en delmängd av instrumenten men framför allt kunna aktivera funktioner på båten. Ett sådant scenario är att kunna starta en av kamerorna i båten remote, via en dashboard eller en app.

Men för att över huvud taget kunna göra det så måste vi kunna kommunicera med WDU:n via utrustning som finns ombord på båten – Raspberry PI..

Ett business-case som ligger högt på agendan är att kunna skicka en signal till en båt om att aktivera en av kamerorna så att en sjöräddningscentral kan se vad som händer (och t.ex. skicka videolänken vidare till en båt som är på väg att ansluta, eller helt enkelt upptäcka att något gått fel och att en ny räddningsinsats behöver sättas in). Tekniskt är det här inte svårare än vad som finns förberett i gränssnittet som ni har tillgång till – att kunna ”klicka på en knapp”.  
Vi behöver också få bekräftat att vårt kommando (knapptryck av något slag) ger det resultat vi väntar oss – har värdet blivit satt eller förväntat resultat utfört?

SSRS har själva labbat lite med att använda node-red för att sätta ihop logik kring vad som skall ske givet olika event, de har aldrig försökt eller lyckats med att skicka en instruktion till en WDU annat än via en browser.   
  
I förlängningen

finns en vision om att kunna ge ut behörighet i en app till en besättning runt en båt så att de kan övervaka och aktivera funktioner de ansvarar för. Samt att centralt ska en grupp människor kunna ha övergripande kontroll över hela flottan.

För att kunna förlita sig på lösningen – efter att vi (ni) knäckt koden så behöver båten rapportera in att den faktiskt är online regelbundet (och inte tappat nät eller att något gått fel).

Applikationen som kör på Raspberry måste också i förlängningen kunna hantera att det finns [mer än en html-sida på WDU](#_WDU_kan_ha) och kunna identifiera vilken sida som är aktuell för integrationen.

Men allt startar här. Lycka till!