Windigipet 2025 – Journey-Planner and MQTT Integration

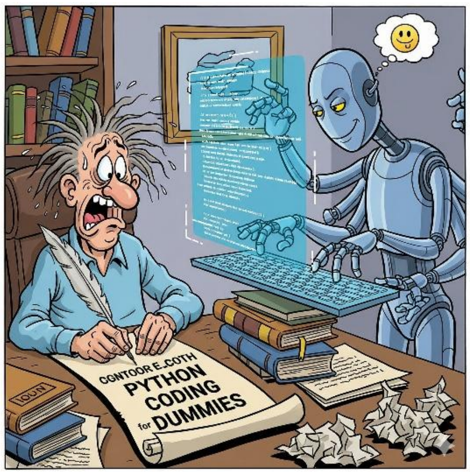
14-9-2025 – Jens Krogsgaard, Copenhagen - Denmark

Table of contents

1. Summary

My big passion is model railways – especially running them with the latest version of Windigipet 2025 – and at the same time, I enjoy coding more or less complex systems.

These days, of course, AI has become the big buzzword everywhere. Being retired, I no longer have colleagues to work with – so I thought I'd try teaming up with ChatGPT on a project. The idea is that I describe the requirements and keep testing the code along the way – while ChatGPT acts as my assistant, doing the coding and advising me on best practices in Python, a language I don't have that much experience with yet.



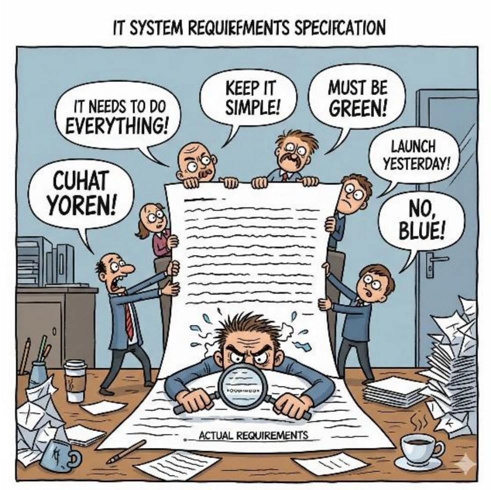
2. Requirements specification

Before I get started together with ChatGPT, we first need to figure out what the program should actually be able to do. It should be clear and easy to follow, but at the same time there should also be a bit of a challenge so it doesn't get boring.

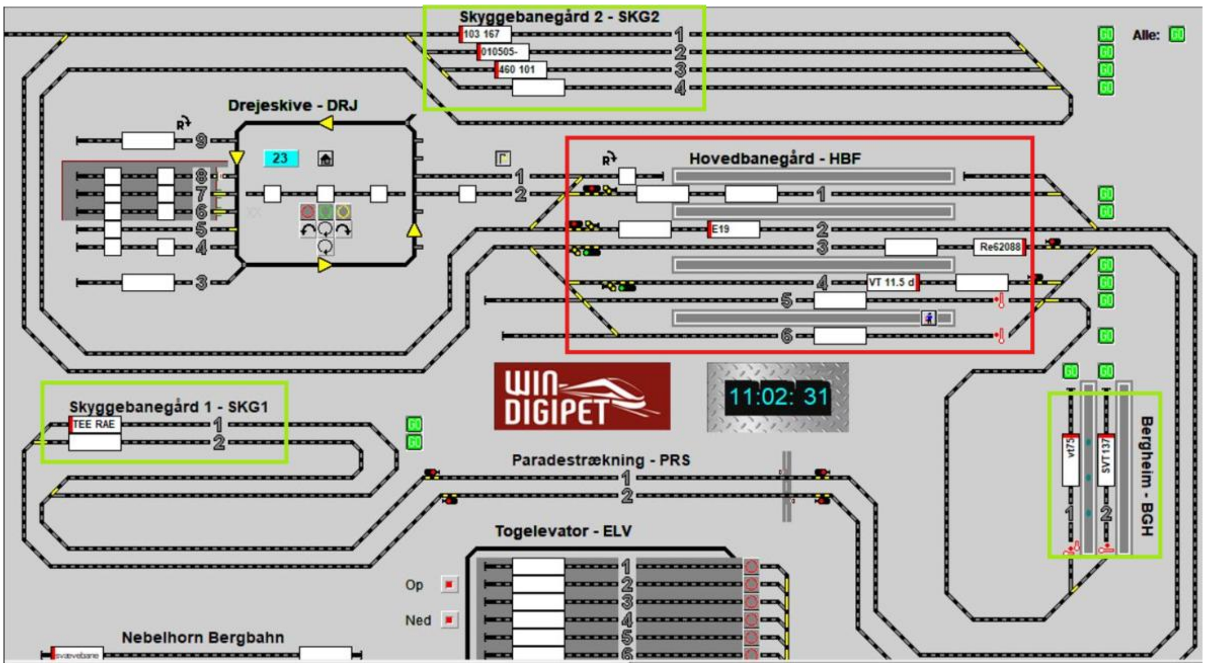
How about creating an Arrivals and Departures board for each of the 4 stations on my model railway – with trains, times, and everything running in real time, of course.

Here are the requirements:

- Create an Arrivals and Departures board for each station on a Windigipet-controlled model railway.
- The number of stations should be configurable so others can also use the program.
- The program should be able to run on the same PC where Windigipet is running.
- There should be an installation kit so that anyone with WDP can install it easily.
- Data is sent from Windigipet to the application via MQTT, which is now supported in the latest version of Windigipet.
- On the Arrivals board for each station, show:
    - track number
    - from-station
    - arrival time
    - train
    - picture of train
- On the Departures board for each station, show:
    - from-track
    - to-station
    - departure time
    - train
    - picture of the train
- There should be language support for Danish, English, and German.
- It should be possible to connect securely to the MQTT broker.



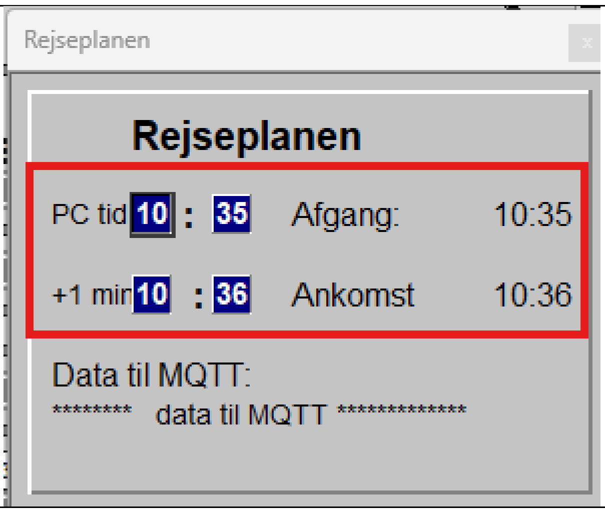
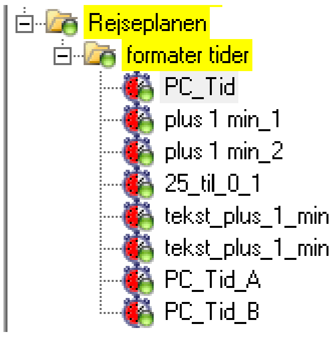
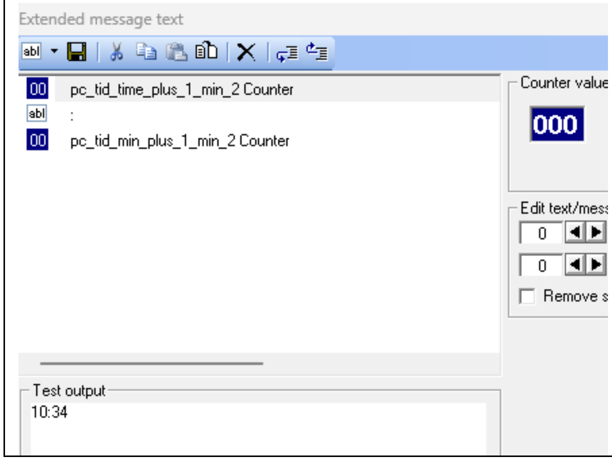
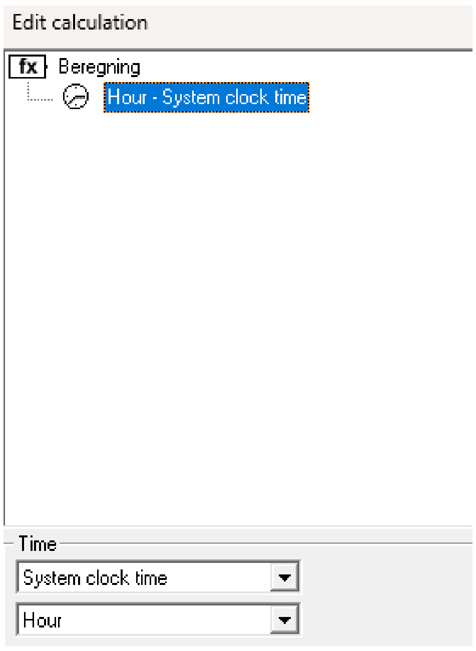
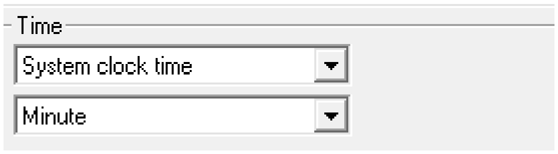
3. We set the scene



I have a main station HBF with 6 tracks, and from here, trains arrive at and depart for two staging yards, SKG1 (2 tracks) and SKG2 (4 tracks), as well as to Bergheim station BGH – 2 tracks.

In Windigipet, I have an automation – a “round trip” – that can control up to 9 train sets at the same time. They run between my 4 stations, and it’s in this context that we’ll be creating our timetable.

4. Changes to Windigipet project

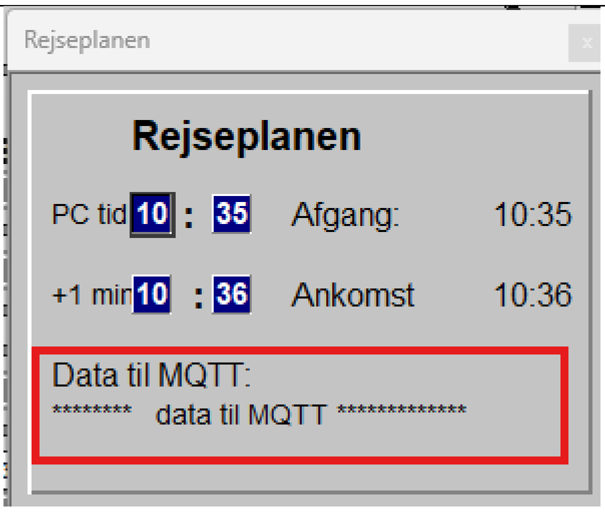
In Windigipet, we'll make a small MultiPlan for our timetable. For departure times, we'll use the PC time, and for arrival times, we'll just take the PC time and add one minute. Of course, that's not completely accurate – but it works pretty well on my layout. And naturally, it can be adjusted as needed.	 Rejseplanen **Rejseplanen** PC tid 10 : 35 Afgang: 10:35 +1 min 10 : 36 Ankomst 10:36 Data til MQTT: ******** data til MQTT *************
The times are calculated in the Dispatcher. First, I split hours and minutes into separate counters – and then I combine them into a text field. Now they're ready to be used.  Rejseplanen formater tider PC_Tid plus 1 min_1 plus 1 min_2 25_til_0_1 tekst_plus_1_min tekst_plus_1_min PC_Tid_A PC_Tid_B combine Counters into a text field  Extended message text pc_tid_time_plus_1_min_2 Counter : pc_tid_min_plus_1_min_2 Counter Counter value 000 Edit text/mess 0 0 Remove s Test output 10:34	For Hour - counter  Edit calculation Beregning Hour - System clock time Time System clock time Hour And for minutes counter:  Time System clock time Minute

We now need to send the following 7 pieces of information to MQTT every time a train leaves one of my 4 stations:

- FromStation
- FromTrack
- ToStation
- ToTrack
- Departure
- Arrive
- Train

The data should be combined into a text string and passed into the field **** Data to MQTT ****.

- Example of a text string: HBF;5;BGH;1;10:32;10:33;SVT137



Rejseplanen

Rejseplanen

PC tid 10 : 35 Afgang: 10:35

+1 min 10 : 36 Ankomst 10:36

Data til MQTT:

******** data til MQTT *************

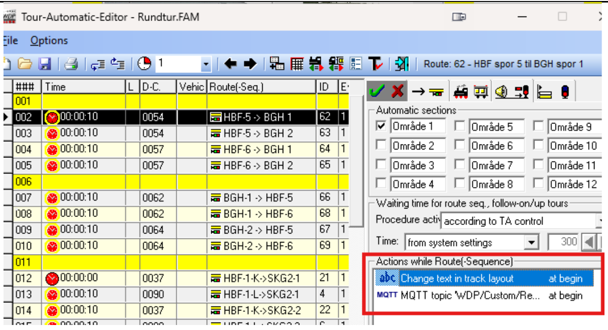
In the Tour Automatic Editor, we have one line for each combination of:

- FromStation
- FromTrack
- ToStation
- ToTrack.

That's where we can put together the information we want and send it to MQTT.

We do this with 2 actions:

- create the string
- send it to MQTT.



Tour-Automatic-Editor - Rundtur.FAM

File Options

Route: 62 - HBF spor 5 til BGH spor 1

###	Time	L	D-C.	Vehic	Route(-Seq.)	ID	E
001							
002	00:00:10		0054		HBF-5 -> BGH 1	62	1
003	00:00:10		0054		HBF-5 -> BGH 2	63	1
004	00:00:10		0057		HBF-6 -> BGH 1	64	1
005	00:00:10		0057		HBF-6 -> BGH 2	65	1
006							
007	00:00:10		0062		BGH-1 -> HBF-5	66	1
008	00:00:10		0062		BGH-1 -> HBF-6	68	1
009	00:00:10		0064		BGH-2 -> HBF-5	67	1
010	00:00:10		0064		BGH-2 -> HBF-6	69	1
011							
012	00:00:00		0037		HBF-1-K->SKG2-1	21	1
013	00:00:10		0090		HBF-1-L->SKG2-1	4	1
014	00:00:10		0037		HBF-1-K->SKG2-2	22	1

Automatic sections

[✓] Område 1 [ ] Område 5 [ ] Område 9

[ ] Område 2 [ ] Område 6 [ ] Område 10

[ ] Område 3 [ ] Område 7 [ ] Område 11

[ ] Område 4 [ ] Område 8 [ ] Område 12

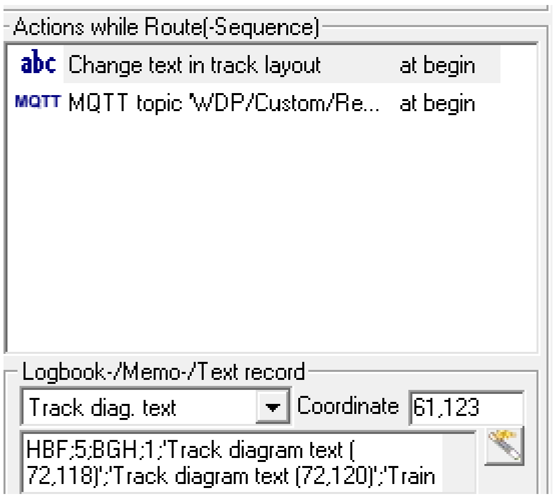
Waiting time for route seq., follow-on/up tours

Procedure acti: according to TA control

Time: from system settings 300

Actions while Route(-Sequence)

Change text in track layout	at begin
MQTT topic 'WDP/Custom/Re...	at begin



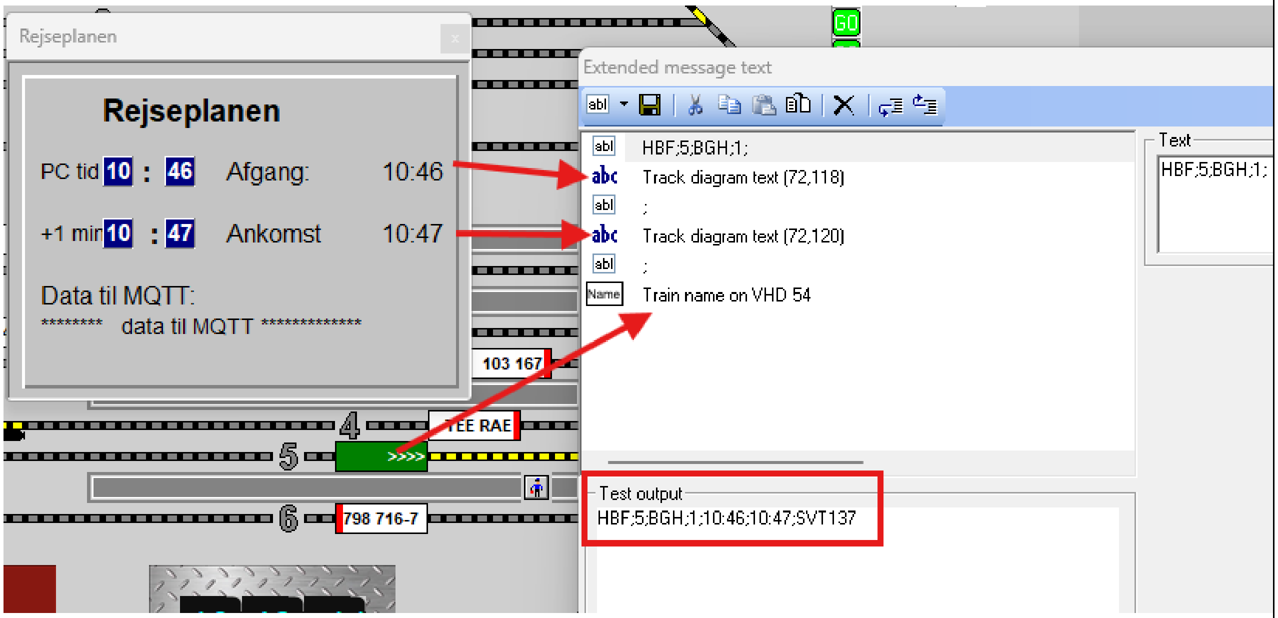
Actions while Route(-Sequence)

Change text in track layout	at begin
MQTT topic 'WDP/Custom/Re...	at begin

Logbook-/Memo-/Text record

Track diag. text Coordinate 61,123

HBF;5;BGH;1;'Track diagram text (72,118)';'Track diagram text (72,120)';'Train


Rejseplanen

PC tid 10 : 46 Afgang: 10:46

+1 min 10 : 47 Ankomst 10:47

Data til MQTT:

******** data til MQTT ************

Extended message text

```
HBF;5;BGH;1;
Track diagram text (72,118)
;
Track diagram text (72,120)
;
Train name on VHD 54
```

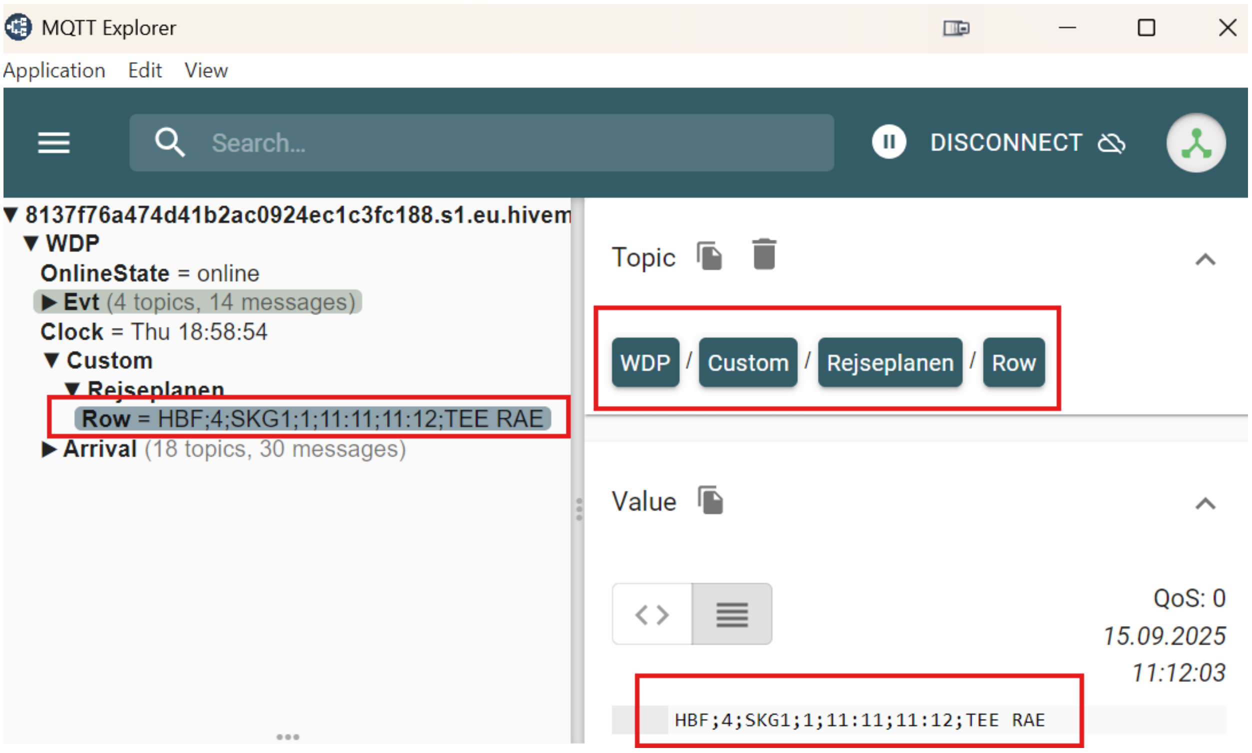
Text

HBF;5;BGH;1;

Test output

```
HBF;5;BGH;1;10:46;10:47;SVT137
```

We use the Extended Text Editor to collect the required information into a text string. The individual elements are separated by ";"
The first part – FromStation, FromTrack, ToStation, ToTrack – comes directly from the row we've selected, so we just type that in, e.g. HBF;5;BGH;1;.
The departure and arrival times we pull from our MultiPlan, while the train name comes from the track plan.
We now have the finished text, and it's created the moment this line in the automation is triggered. It then needs to be sent straight to MQTT.


MQTT Explorer

Application Edit View

```
▼ 8137f76a474d41b2ac0924ec1c3fc188.s1.eu.hivem
  ▼ WDP
    OnlineState = online
    ▶ Evt (4 topics, 14 messages)
    Clock = Thu 18:58:54
    ▼ Custom
      ▼ Rejseplanen
        Row = HBF;4;SKG1;1;11:11;11:12;TEE RAE
    ▶ Arrival (18 topics, 30 messages)
```

Topic

WDP / Custom / Rejseplanen / Row

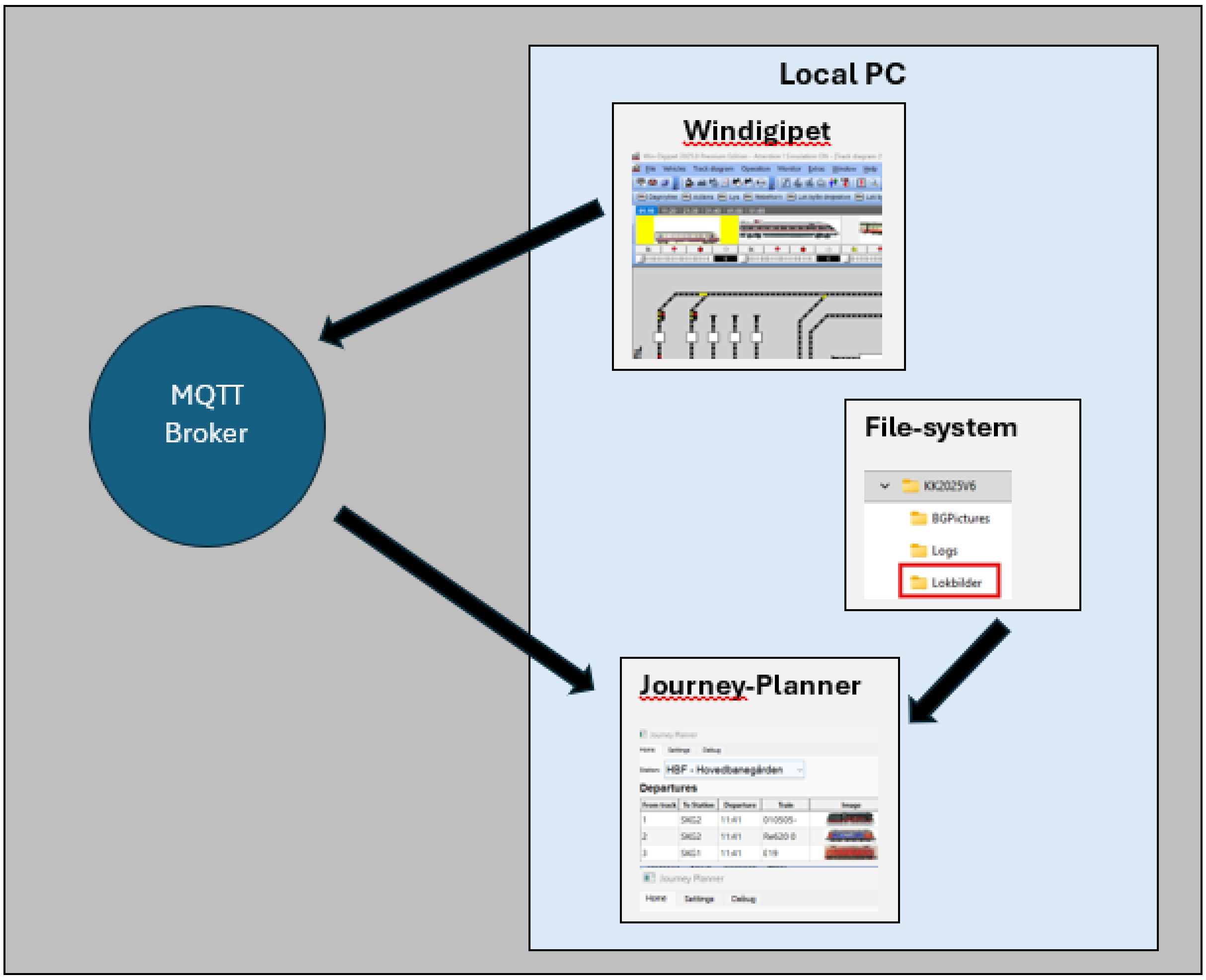
Value

QoS: 0
15.09.2025
11:12:03

```
HBF;4;SKG1;1;11:11;11:12;TEE RAE
```

In an MQTT explorer, we can now see that data is being received from Windigipet. We always see the latest message, and the next step is to create a program that reads these messages and stores them in a table. So, we're done making changes in Windigipet and can now start building the new program, *Journey-planner*.

5. Dataflow



Here you can see the data flow for our new system. I'm running my MQTT broker in the cloud – but it could just as well be installed on a local PC. From Windigipet, messages are sent to the MQTT broker, and these are then picked up by our new Windows app *Journey Planner*.

We also want to show pictures of the locomotives on our Arrivals and Departures boards. We can do that by pulling the images from the file system – in the *Vehicles* and *Lokbilder* folders under the current Windigipet project.

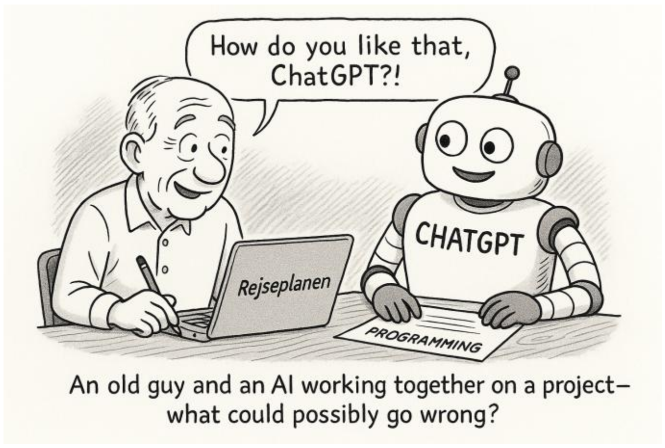
6. Programming the Journey Planner

🚂 **How *Rejseplanen* was built – Jens & ChatGPT's Adventure**

Sketch 1 – *The Big Idea*

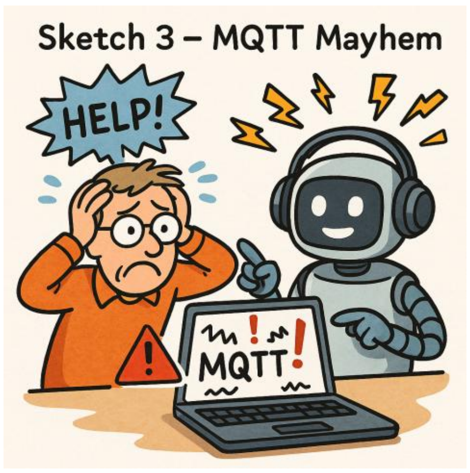
Every great project starts with an idea. Jens had the trains, but he wanted a smart **Windows app** to plan departures and arrivals.

Sketch 2 – *The Unlikely Partnership*


How do you like that, ChatGPT?!
Rejseplanen
CHATGPT
PROGRAMMING
An old guy and an AI working together on a project– what could possibly go wrong?

Jens set the requirements, ChatGPT wrote the code. One explained the railway logic, the other translated it into Python and PyQt.

Sketch 3 – *MQTT Mayhem*

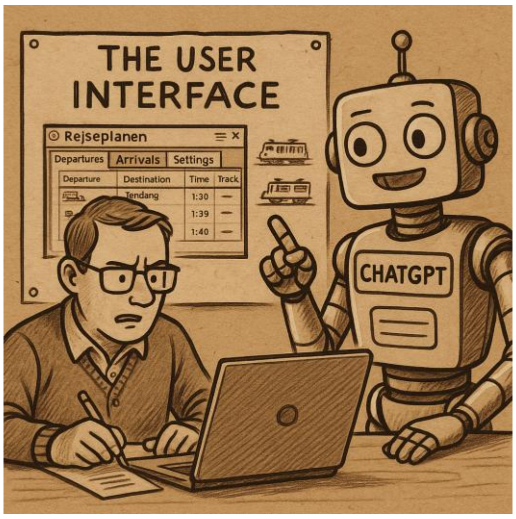


We connected to MQTT – the trains started talking. Data flew in, and our tables began to live.

Sketch 4 – *Debugging Adventures*

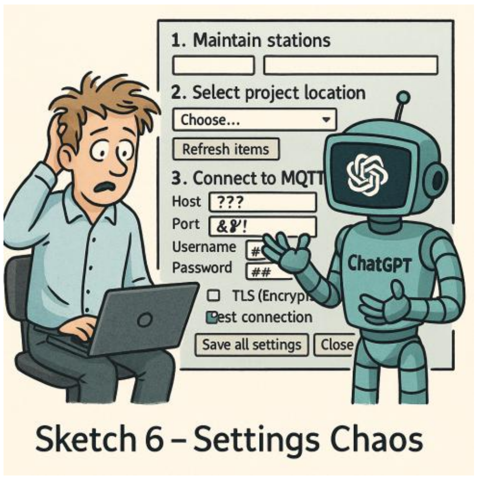
There were… a few duplicates. But after some logic, we found the right keys: **FromStation + FromTrack** (and later ToStation + ToTrack).

Sketch 5 – *The User Interface*



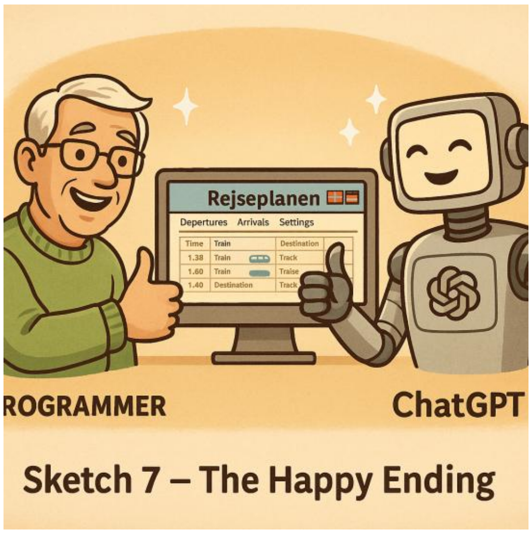
The front page showed departures and arrivals, complete with locomotive pictures pulled straight from the Windigipet project.

Sketch 6 – *Settings Chaos*



The settings page became a control center: stations, MQTT config, project folder, and even language selection with cute flag buttons.

Sketch 7 – *The Happy Ending*



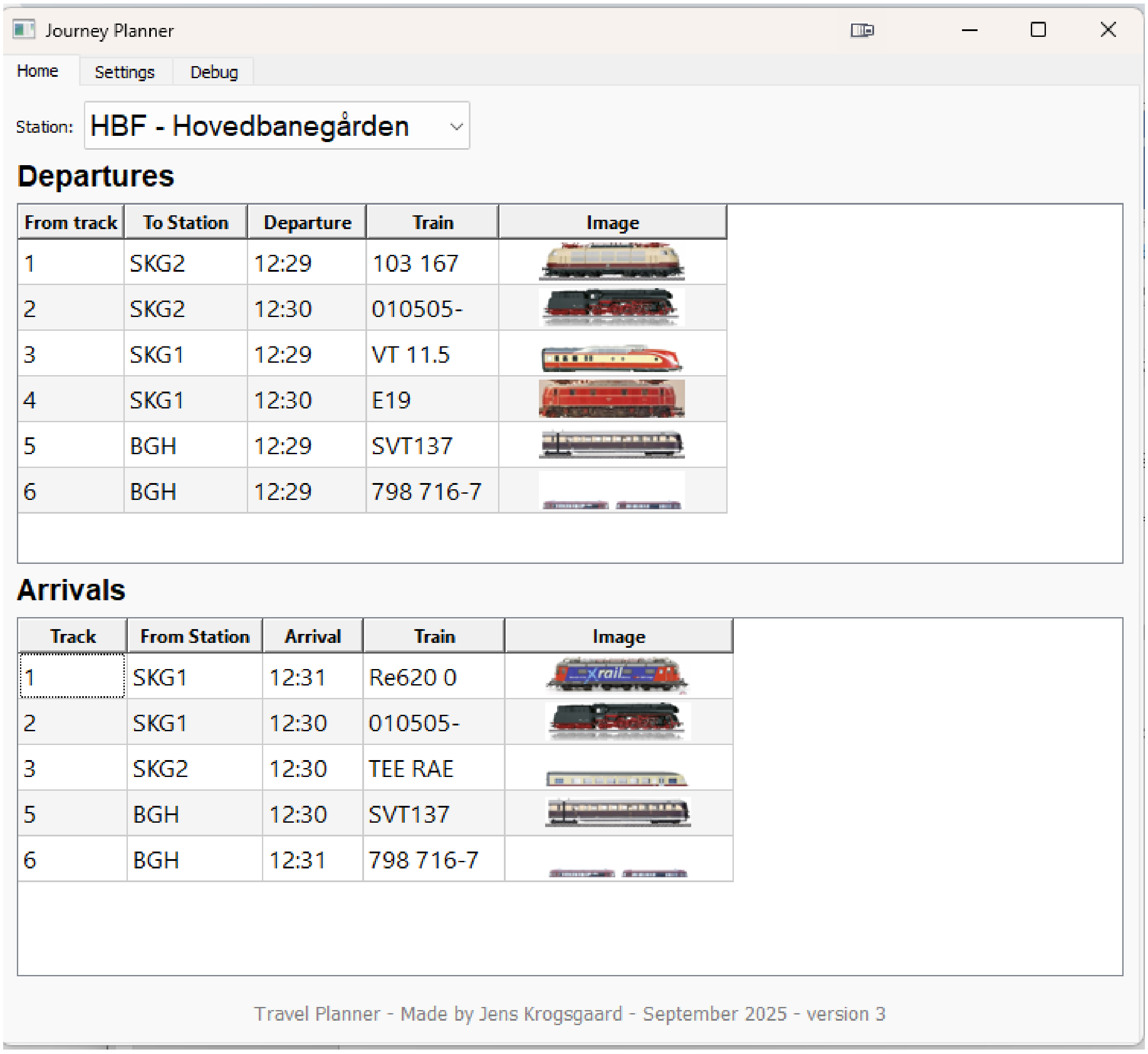
After many iterations (and some late-night debugging), the app was ready to run on the real railway. Packed as an .exe, with flags, icons, and all.

👉 That's the story of *Rejseplanen*:
One user with a clear vision + one AI with endless patience = one working app.

So how did the program actually get built?
Well, of course, I asked my buddy ChatGPT — and it came back with the points you just saw. And honestly, that's pretty accurate: I did the specifying and testing, while ChatGPT did the coding and kept throwing in useful suggestions along the way.

I picked up a lot during the process — every now and then I even tweaked the code myself and fed the changes back to ChatGPT.

7. Application Journey Planner



Station: HBF - Hovedbanegården

Departures

From track	To Station	Departure	Train	Image
1	SKG2	12:29	103 167	
2	SKG2	12:30	010505-	
3	SKG1	12:29	VT 11.5	
4	SKG1	12:30	E19	
5	BGH	12:29	SVT137	
6	BGH	12:29	798 716-7	

Arrivals

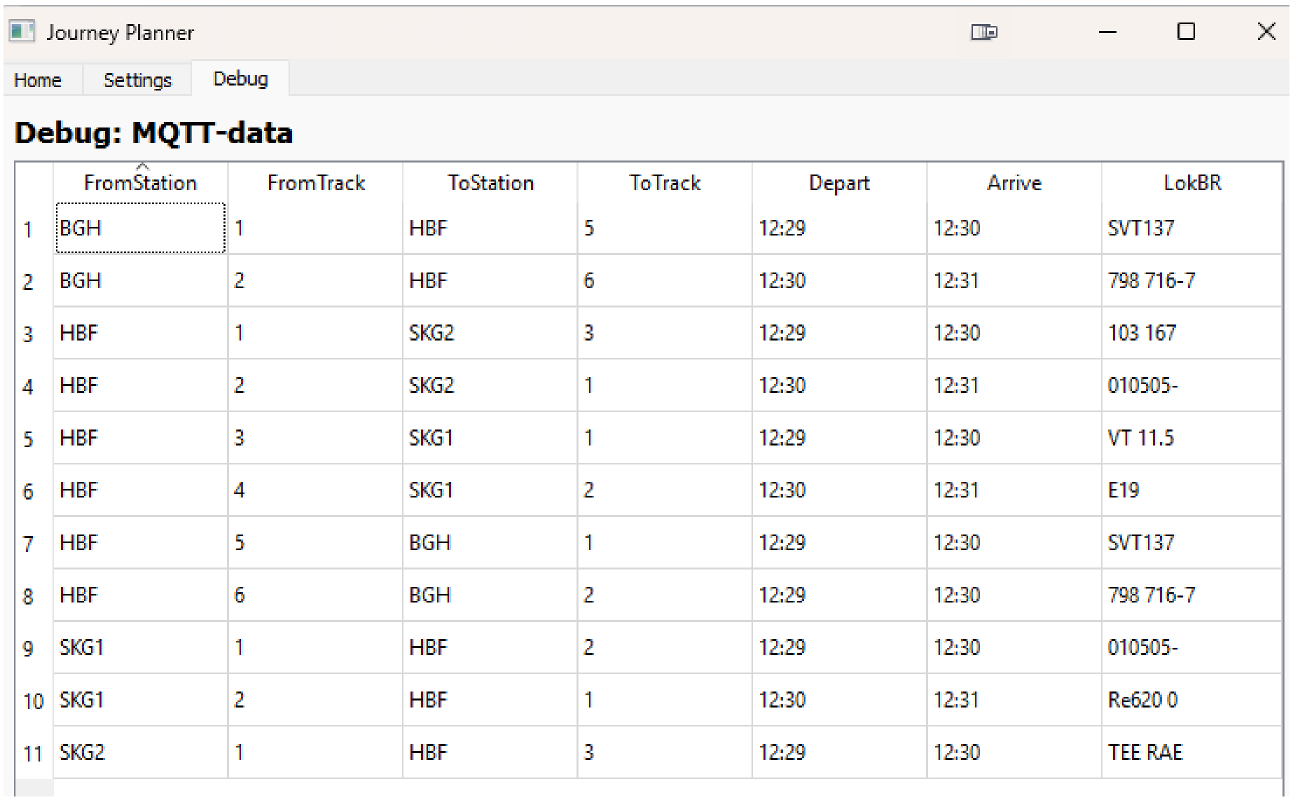
Track	From Station	Arrival	Train	Image
1	SKG1	12:31	Re620 0	
2	SKG1	12:30	010505-	
3	SKG2	12:30	TEE RAE	
5	BGH	12:30	SVT137	
6	BGH	12:31	798 716-7	

Travel Planner - Made by Jens Krogsgaard - September 2025 - version 3

Here’s the finished app.

It’s got three tabs, which we’ll walk through in detail below.

7.1. Debug - Tab



Debug: MQTT-data

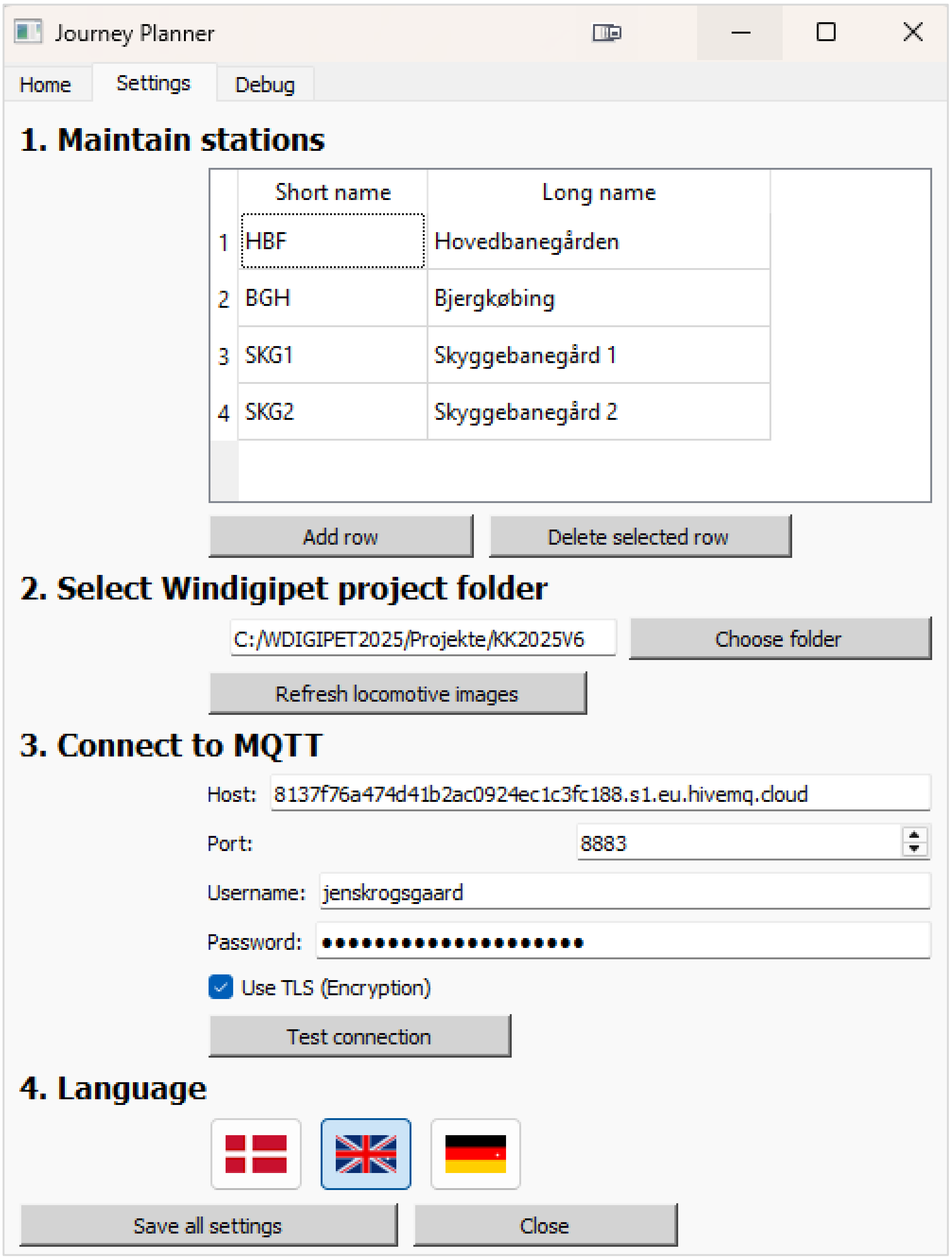
	FromStation	FromTrack	ToStation	ToTrack	Depart	Arrive	LokBR
1	BGH	1	HBF	5	12:29	12:30	SVT137
2	BGH	2	HBF	6	12:30	12:31	798 716-7
3	HBF	1	SKG2	3	12:29	12:30	103 167
4	HBF	2	SKG2	1	12:30	12:31	010505-
5	HBF	3	SKG1	1	12:29	12:30	VT 11.5
6	HBF	4	SKG1	2	12:30	12:31	E19
7	HBF	5	BGH	1	12:29	12:30	SVT137
8	HBF	6	BGH	2	12:29	12:30	798 716-7
9	SKG1	1	HBF	2	12:29	12:30	010505-
10	SKG1	2	HBF	1	12:30	12:31	Re620 0
11	SKG2	1	HBF	3	12:29	12:30	TEE RAE

This is the Debug tab.
It's not something you'd normally use — it's mainly there for testing the data transfer. The table shows the data coming in from Windigipet.

Whenever a new row arrives, the program checks if any of the combinations *FromStation* + *FromTrack* or *ToStation* + *ToTrack* already exist. If so, the old rows are deleted and the new one is inserted.

So this screen is super handy for testing the data flow — and at the same time, the table forms the basis for generating the arrivals and departures boards on the front page.

7.2. Settings - Tab



1. Maintain stations

	Short name	Long name
1	HBF	Hovedbanegården
2	BGH	Bjergkøbing
3	SKG1	Skyggebanegård 1
4	SKG2	Skyggebanegård 2

2. Select Windigipet project folder

C:/WDIGIPET2025/Projekte/KK2025V6

3. Connect to MQTT

Host: 8137f76a474d41b2ac0924ec1c3fc188.s1.eu.hivemq.cloud
Port: 8883
Username: jenskrogsgaard
Password: ••••••••••••••••••••
[✓] Use TLS (Encryption)

4. Language

First, you create a list of stations from Windigipet.
The *Short Name* corresponds to the name that Windigipet actually sends out.

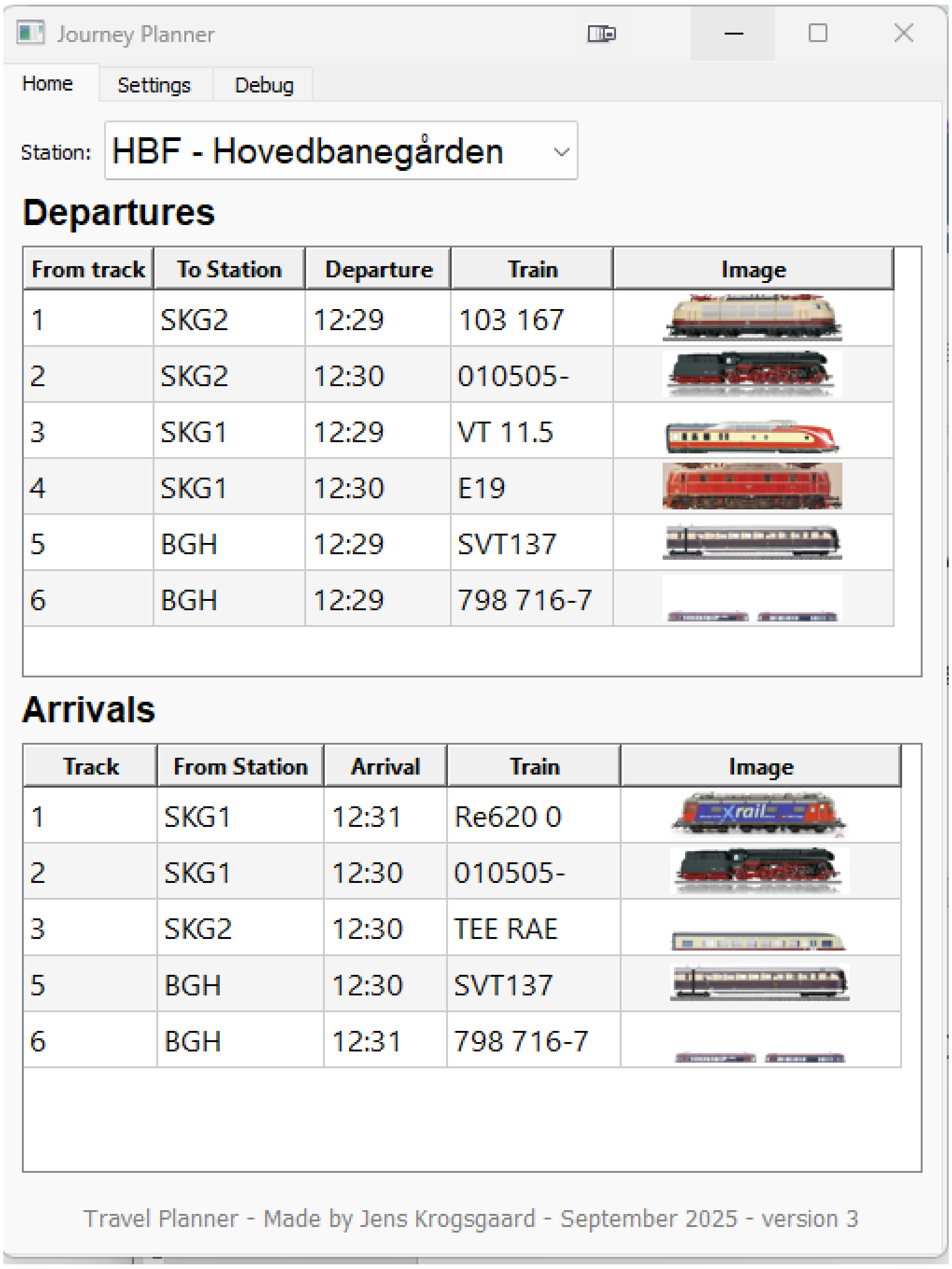
Next, you select the folder where your current Windigipet project is stored. After that, you set up the connection to the same MQTT broker that Windigipet uses — and you can even test the connection to make sure it works.

Finally, you pick a language. All labels and texts in the app will switch to the selected language. Once everything looks good, you hit **Save**.

If you change the language, you'll need to restart the app — just like in Windigipet.

Button **Refresh locomotive images** – this button isn't normally used — it's only there for debugging.

7.3. Home – Tab



Station: HBF - Hovedbanegården

Departures

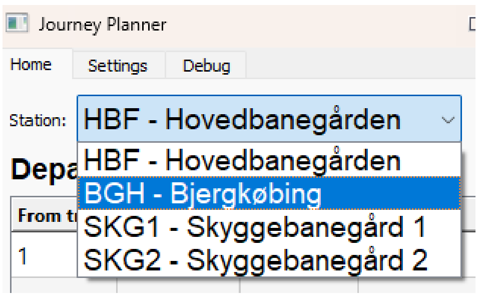
From track	To Station	Departure	Train	Image
1	SKG2	12:29	103 167	
2	SKG2	12:30	010505-	
3	SKG1	12:29	VT 11.5	
4	SKG1	12:30	E19	
5	BGH	12:29	SVT137	
6	BGH	12:29	798 716-7	

Arrivals

Track	From Station	Arrival	Train	Image
1	SKG1	12:31	Re620 0	
2	SKG1	12:30	010505-	
3	SKG2	12:30	TEE RAE	
5	BGH	12:30	SVT137	
6	BGH	12:31	798 716-7	

Travel Planner - Made by Jens Krogsgaard - September 2025 - version 3

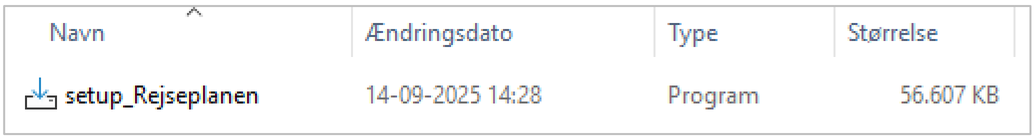
At the top of the screen you pick a station — after that, departures and arrivals for that station are shown.



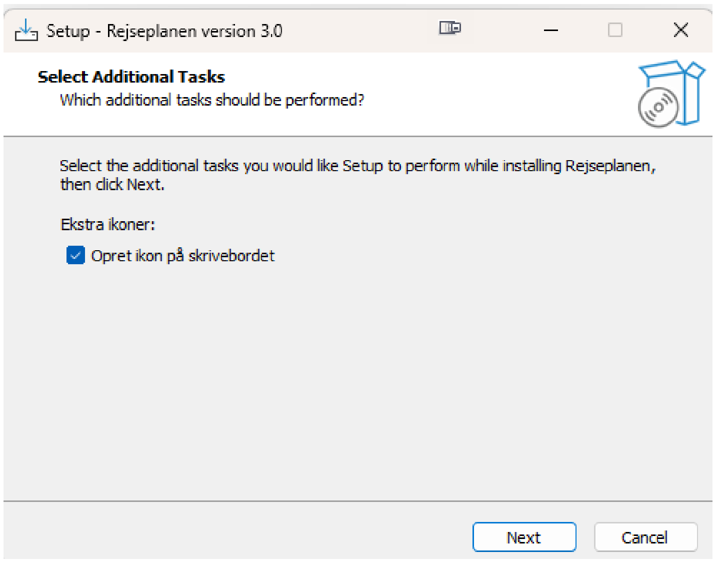
The data is sorted by track number, and the pictures are pulled from Windigipet for the current train.

8. Program distribution – and installation

The program is distributed as a single .exe file. You just run it, and then a normal Windows installation starts.



Navn	Ændringsdato	Type	Størrelse
setup_Rejseplanen	14-09-2025 14:28	Program	56.607 KB



Setup - Rejseplanen version 3.0

Select Additional Tasks
Which additional tasks should be performed?

Select the additional tasks you would like Setup to perform while installing Rejseplanen, then click Next.

Ekstra ikoner:

[✓] Opret ikon på skrivebordet

You can create a desktop shortcut if you like. Then just start the program, tweak the settings a bit, and you're up and running.

I've previously written a guide on how to connect Windigipet to an MQTT broker – you can check it out here: MQTT - Windigipet guide