Production Supporting Systems in Factories

ระบบสนับสนุนการผลิตในโรงงานอุตสาหกรรม

Machine learning

> Image classification

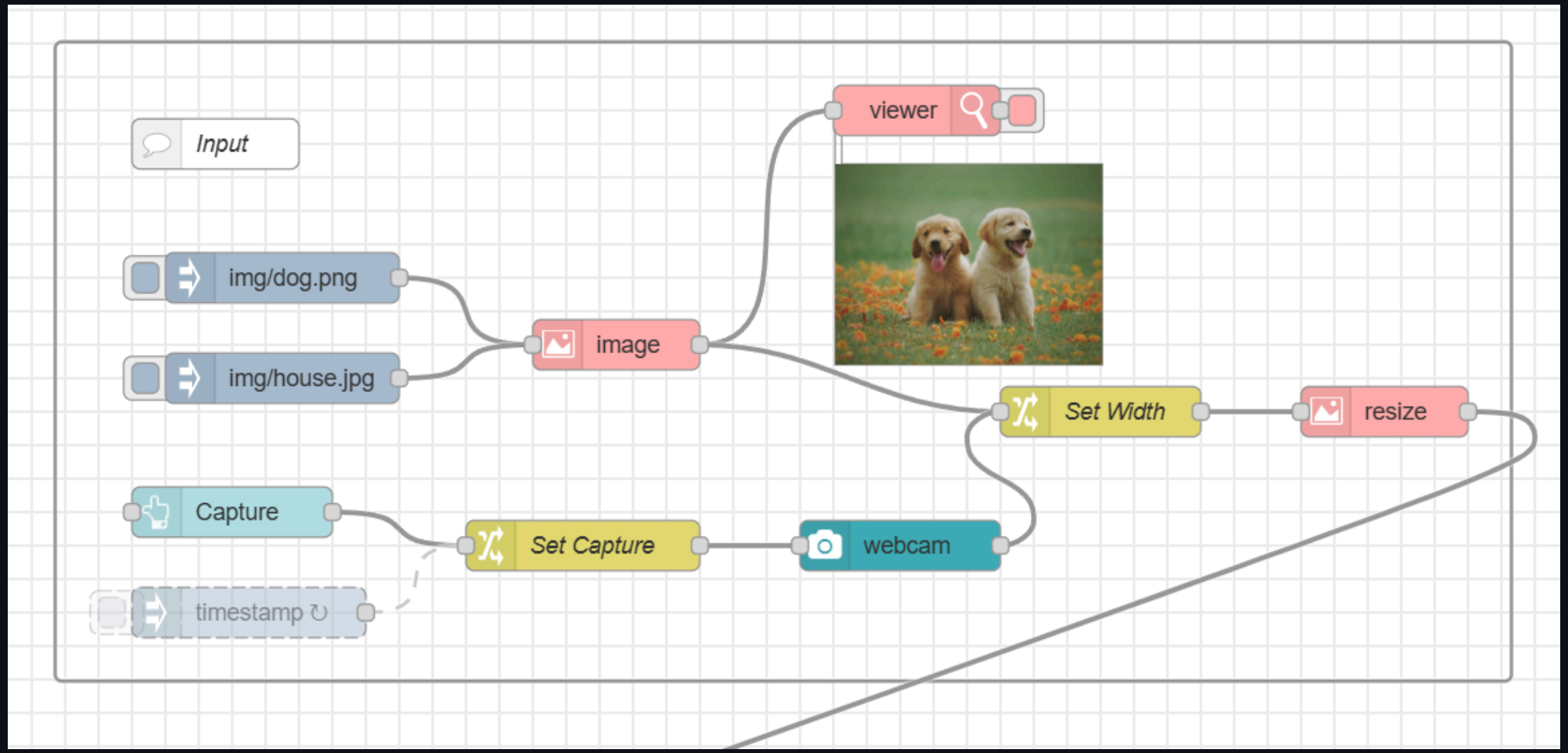
Setting up ML server

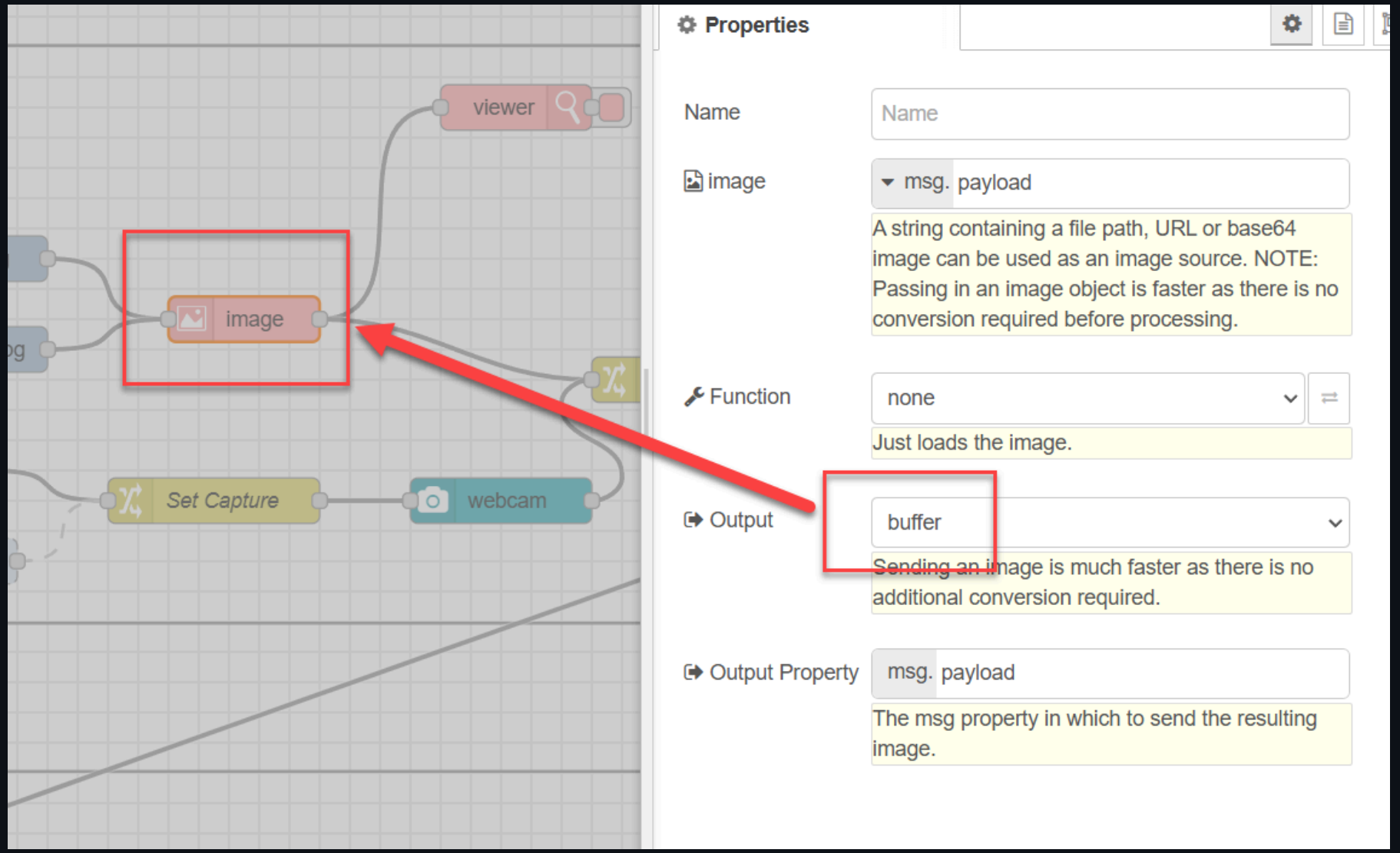
- Get code
- `pnpm install`
- `pnpm run build`
- `pnpm run start`
- Test if server is running.
  - http://localhost:3003

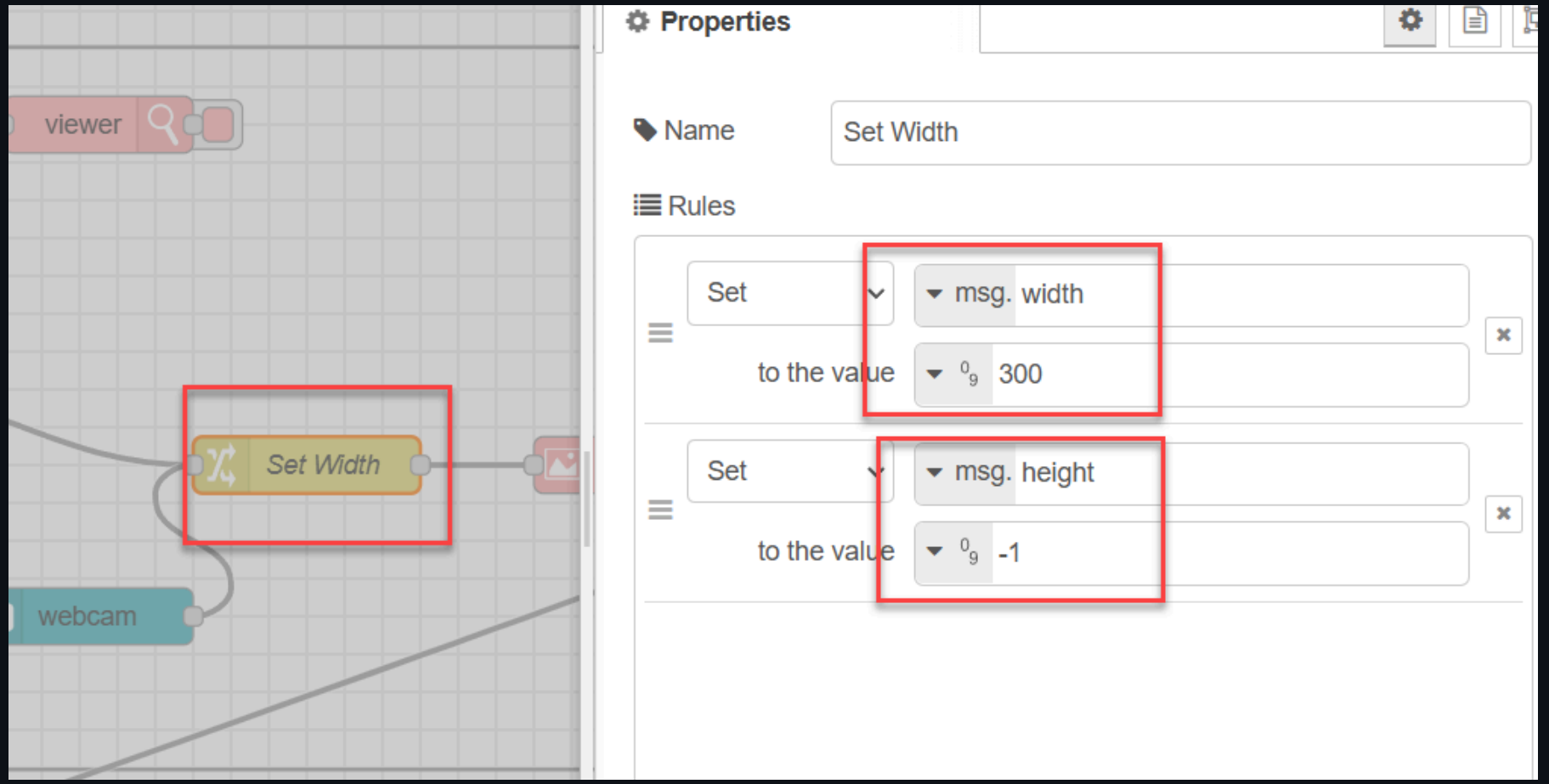
Preparing `node-red`

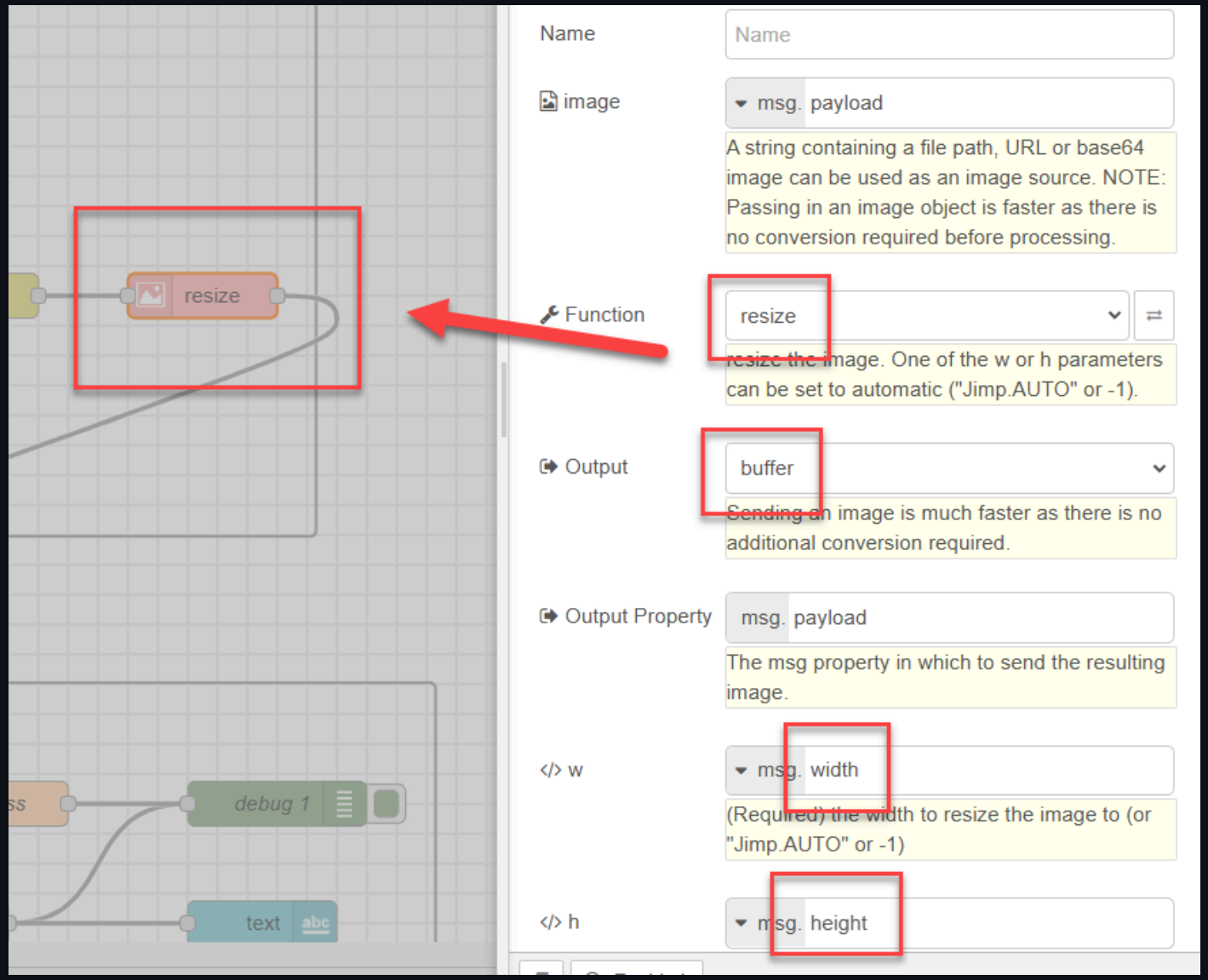
- `pnpm install node-red-contrib-image-tools node-red-node-ui-webcam node-red-dashboard`

Image input flow




viewer
image
Set Capture
webcam
Properties
Name
Name
image
msg. payload
A string containing a file path, URL or base64 image can be used as an image source. NOTE: Passing in an image object is faster as there is no conversion required before processing.
Function
none
Just loads the image.
Output
buffer
Sending an image is much faster as there is no additional conversion required.
Output Property
msg. payload
The msg property in which to send the resulting image.


viewer
Set Width
webcam
Properties
Name
Set Width
Rules
Set
msg. width
to the value
300
Set
msg. height
to the value
-1


resize
debug 1
text
Name
Name
image
msg. payload
A string containing a file path, URL or base64 image can be used as an image source. NOTE: Passing in an image object is faster as there is no conversion required before processing.
Function
resize
resize the image. One of the w or h parameters can be set to automatic ("Jimp.AUTO" or -1).
Output
buffer
Sending an image is much faster as there is no additional conversion required.
Output Property
msg. payload
The msg property in which to send the resulting image.
</> w
msg. width
(Required) the width to resize the image to (or "Jimp.AUTO" or -1)
</> h
msg. height

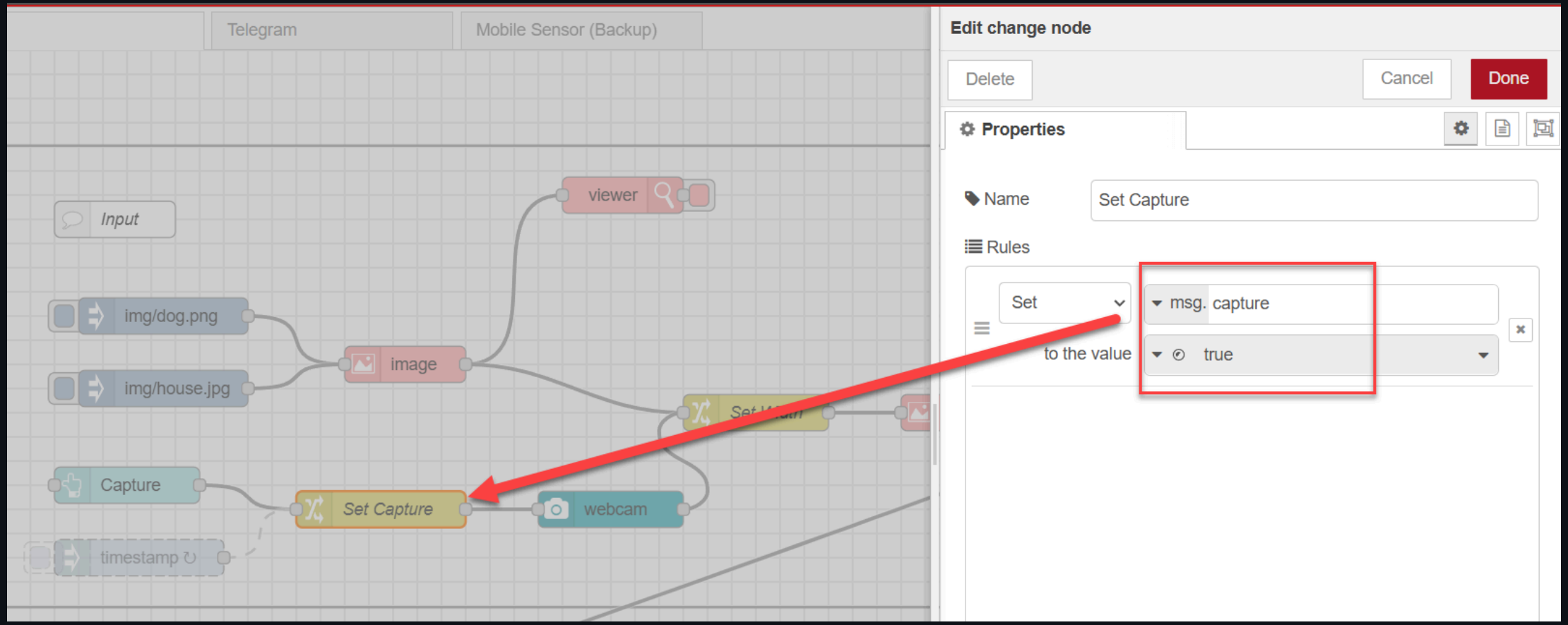
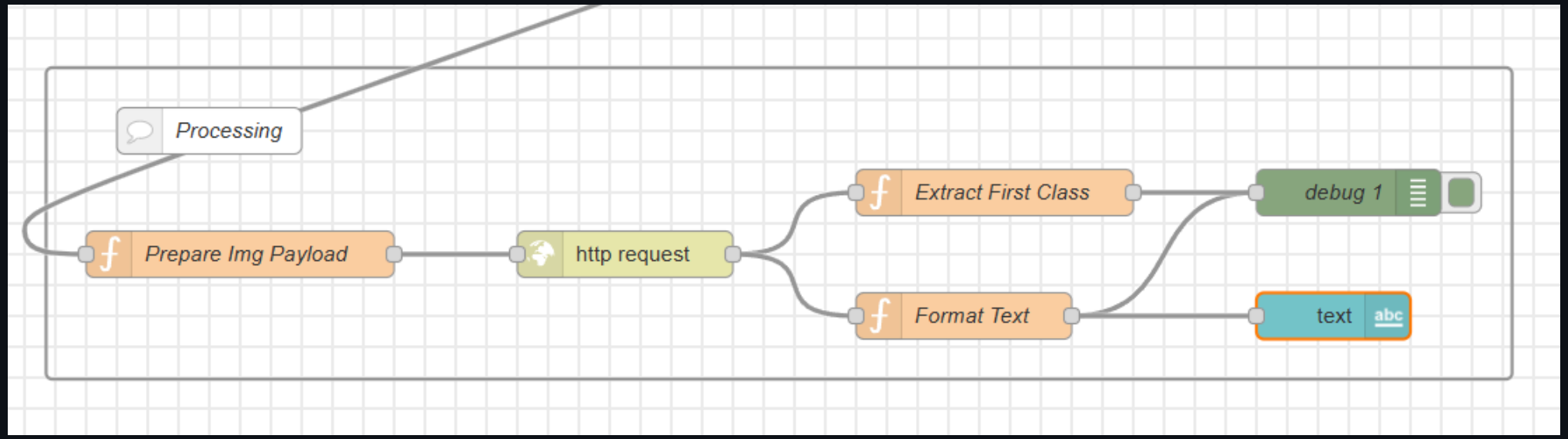

Telegram
Mobile Sensor (Backup)
Input
viewer
img/dog.png
image
img/house.jpg
Capture
Set Capture
webcam
timestamp
Edit change node
Delete
Cancel
Done
Properties
Name
Set Capture
Rules
Set
msg. capture
to the value
true

Image processing flow



`function` node (Prepare Img Payload)

```
msg.payload = { imageEncoded: Buffer.from(msg.payload).toString("base64") };
msg.headers = { "Content-Type": "application/x-www-form-urlencoded" };
msg.url = "http://localhost:3003/upload_base64";
return msg;
```

`function` node (Extract First Class)

```
const obj = JSON.parse(msg.payload);
msg.payload = obj.predictions[0].class;
return msg;
```

`function` node (Format Text)

```
const obj = JSON.parse(msg.payload);
let textOut = "";
for (const pred of obj.predictions) {
  const classStr = pred.class;
  const score = pred.score;
  const scoreP = (score * 100).toFixed(1);
  textOut += `👉${classStr} (${scoreP}%) <br/>`;
}
const dt = new Date();
const datestring = dt.toLocaleDateString();
const timestring = dt.toLocaleTimeString();
textOut += `<br/>📅${datestring} ⏰${timestring}`;
msg.payload = textOut;
return msg;
```