AUTODESK. UNIVERSITY

IoT Manufacturing Demo: Optimizing CNC Toolpaths

Nathan C. Skalsky

Senior Product Manager, Autodesk Forge | @speedbirdpdx

Varun Patil

Developer Technical Consultant, Autodesk Forge

About the speaker

Nathan Skalsky

I’m a senior product manager for Autodesk Forge and a ‘maker-of-things’ at heart. I enjoy enabling workflows and capabilities that enable more efficient expression of human creativity through automation and simplification.

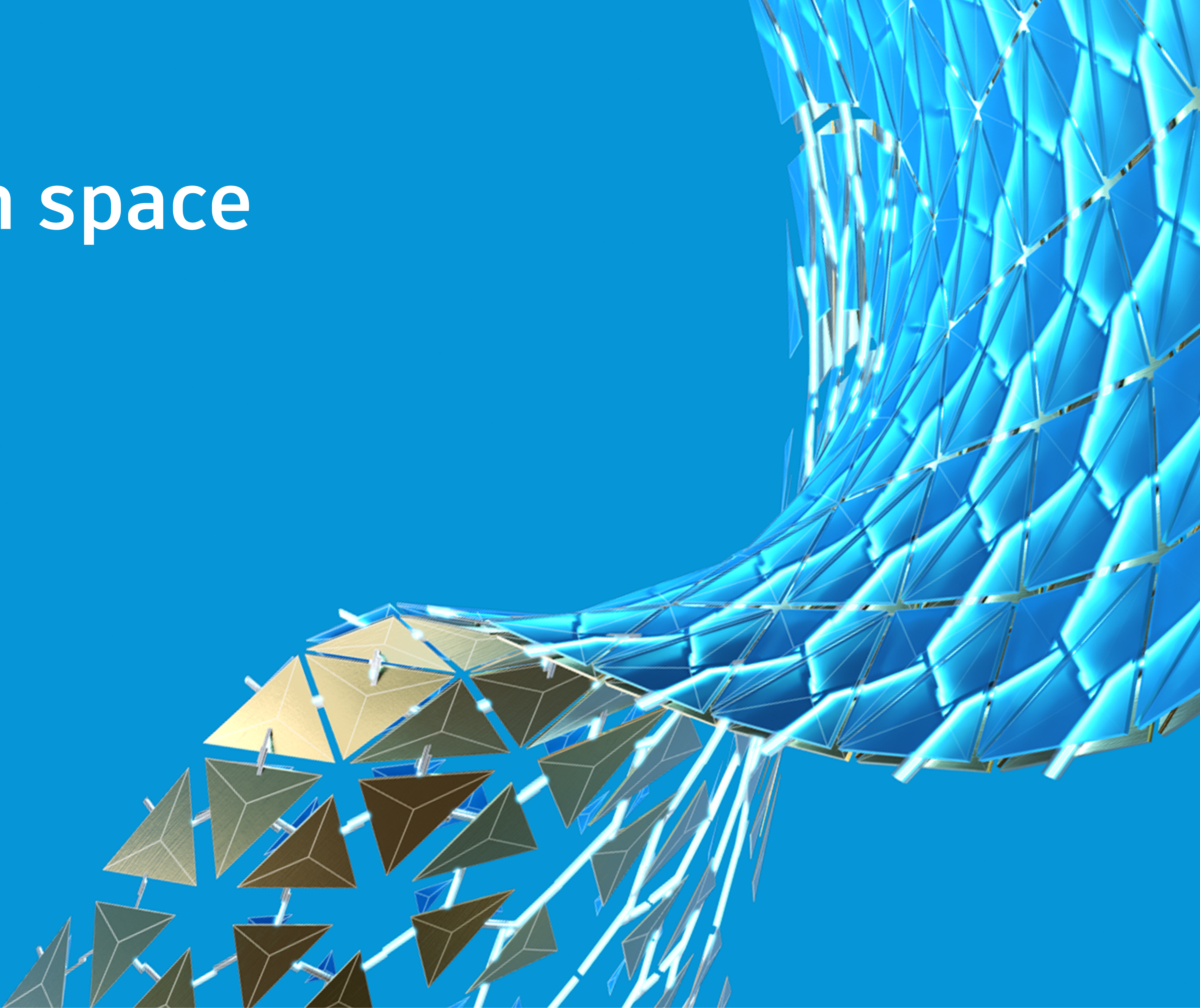
I started my career at IBM designing small parts of big systems, a software development manager for Fusion 360, a couple startups, and happy to be back at Autodesk as a Forge product manager!

About the speaker

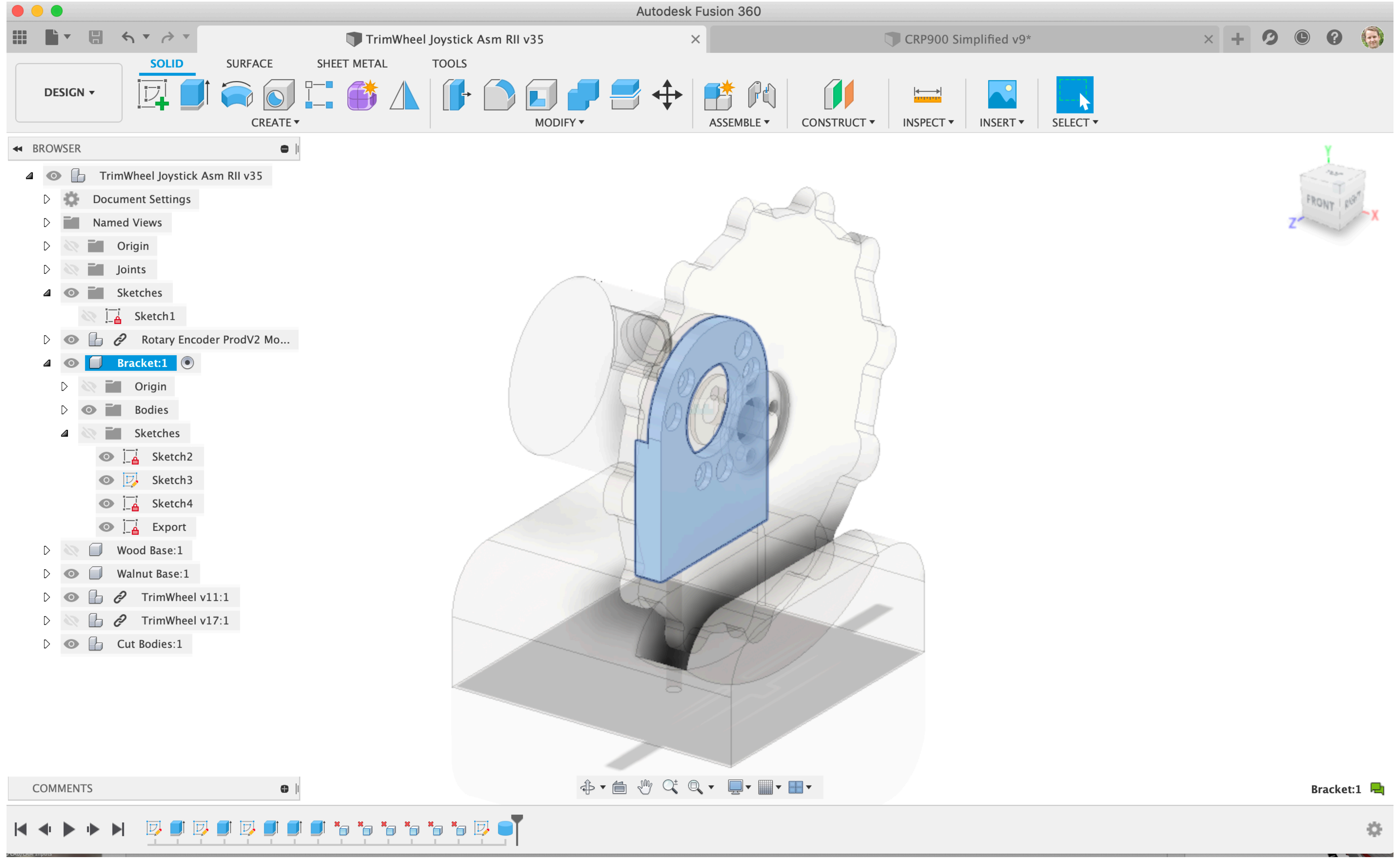
Varun Patil

I’m full stack developer, working in Autodesk for more than two and half years as part of Forge development team. I create sample projects using Forge APIs and help our customers along the way.

The problem space



Making Better Things with Fusion 360, Forge, and CNC Machines!



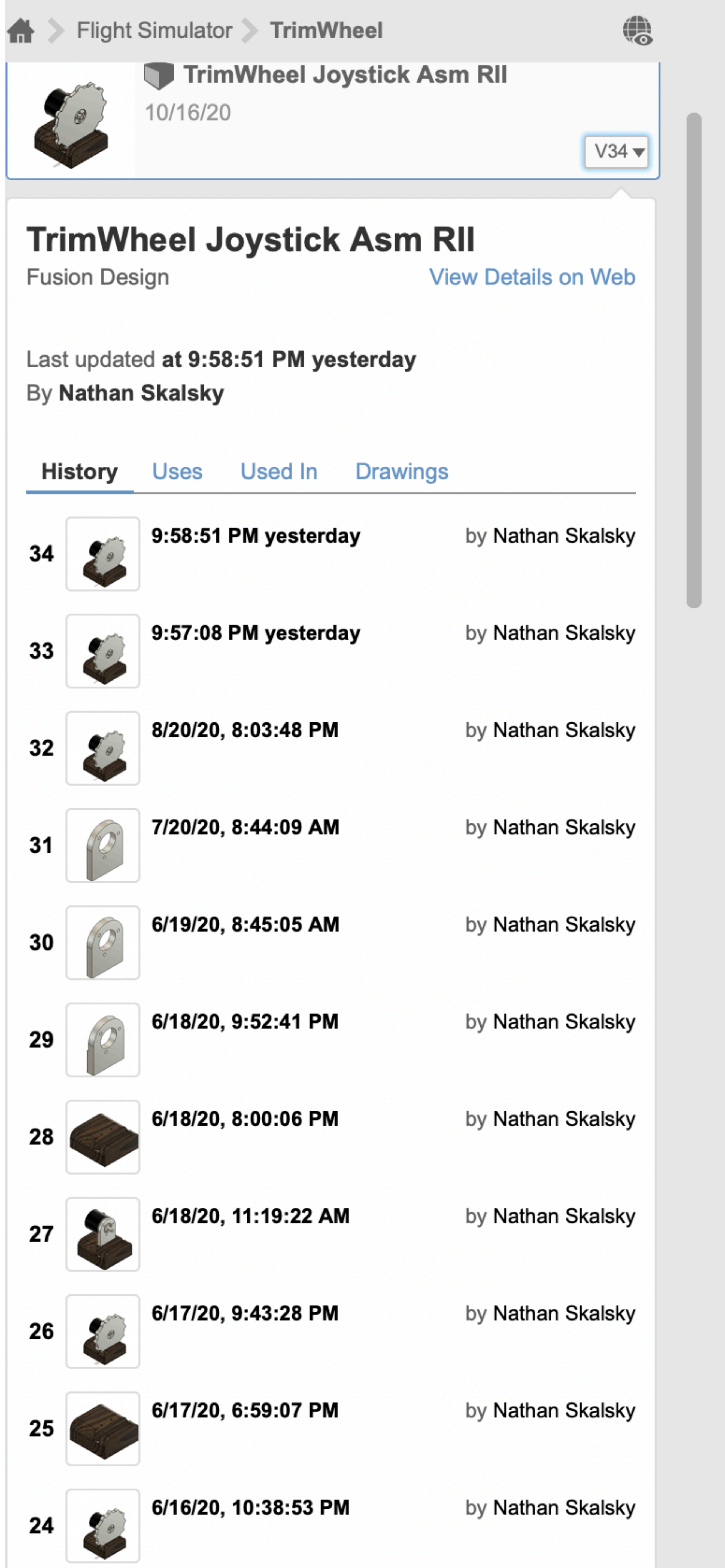
Fusion 360, 3D CAD, CAM and More!

CNC Router (Benchtop PRO 2’x2’)

A Maker’s Dilemma

Design-to-Manufacturing workflows (CNC, 3DP, Laser) are often primarily a one-way process. How to we ensure we learn from our outcomes?

- **Close the loop**, aka ‘light’ Process Control with Forge APIs
- **Learn from outcomes**, capture meaningful results
- **Make better stuff!** The better our workflows become, the better our outcomes will get.



Flight Simulator > TrimWheel

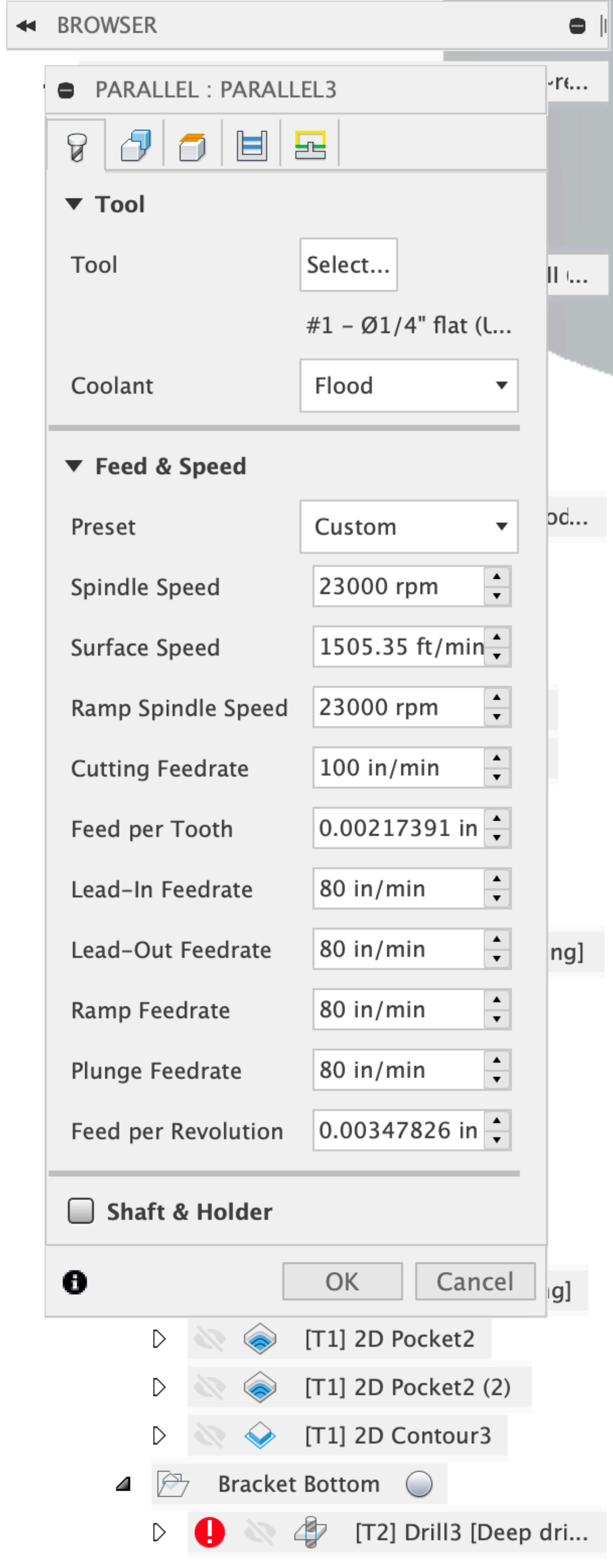
TrimWheel Joystick Asm RII
10/16/20
V34

TrimWheel Joystick Asm RII

Fusion Design
View Details on Web

Last updated **at 9:58:51 PM yesterday**
By **Nathan Skalsky**

34	9:58:51 PM yesterday	by Nathan Skalsky
33	9:57:08 PM yesterday	by Nathan Skalsky
32	8/20/20, 8:03:48 PM	by Nathan Skalsky
31	7/20/20, 8:44:09 AM	by Nathan Skalsky
30	6/19/20, 8:45:05 AM	by Nathan Skalsky
29	6/18/20, 9:52:41 PM	by Nathan Skalsky
28	6/18/20, 8:00:06 PM	by Nathan Skalsky
27	6/18/20, 11:19:22 AM	by Nathan Skalsky
26	6/17/20, 9:43:28 PM	by Nathan Skalsky
25	6/17/20, 6:59:07 PM	by Nathan Skalsky
24	6/16/20, 10:38:53 PM	by Nathan Skalsky



PARALLEL : PARALLEL3

Tool

Tool	Select...
	#1 – Ø1/4" flat (L...
Coolant	Flood

Feed & Speed

Preset	Custom
Spindle Speed	23000 rpm
Surface Speed	1505.35 ft/min
Ramp Spindle Speed	23000 rpm
Cutting Feedrate	100 in/min
Feed per Tooth	0.00217391 in
Lead-In Feedrate	80 in/min
Lead-Out Feedrate	80 in/min
Ramp Feedrate	80 in/min
Plunge Feedrate	80 in/min
Feed per Revolution	0.00347826 in

□ **Shaft & Holder**

[T1] 2D Pocket2
[T1] 2D Pocket2 (2)
[T1] 2D Contour3
Bracket Bottom
[T2] Drill3 [Deep dri...

Connecting Results back to CAD/CAM Inputs

Feedback brainstorm for simplified Process Control

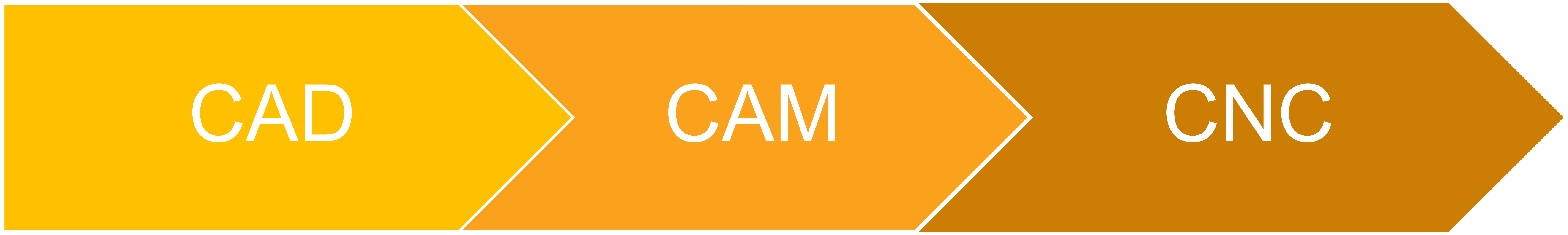
Automated Feedback:

- Machine Vision
- Sound
- > **Temperature**
- Servo/PLC/CMM feedback

Manual Analysis:

- Visual inspection of surface finish
- Machine operator notes
- Engineering functional feedback
- Customer feedback

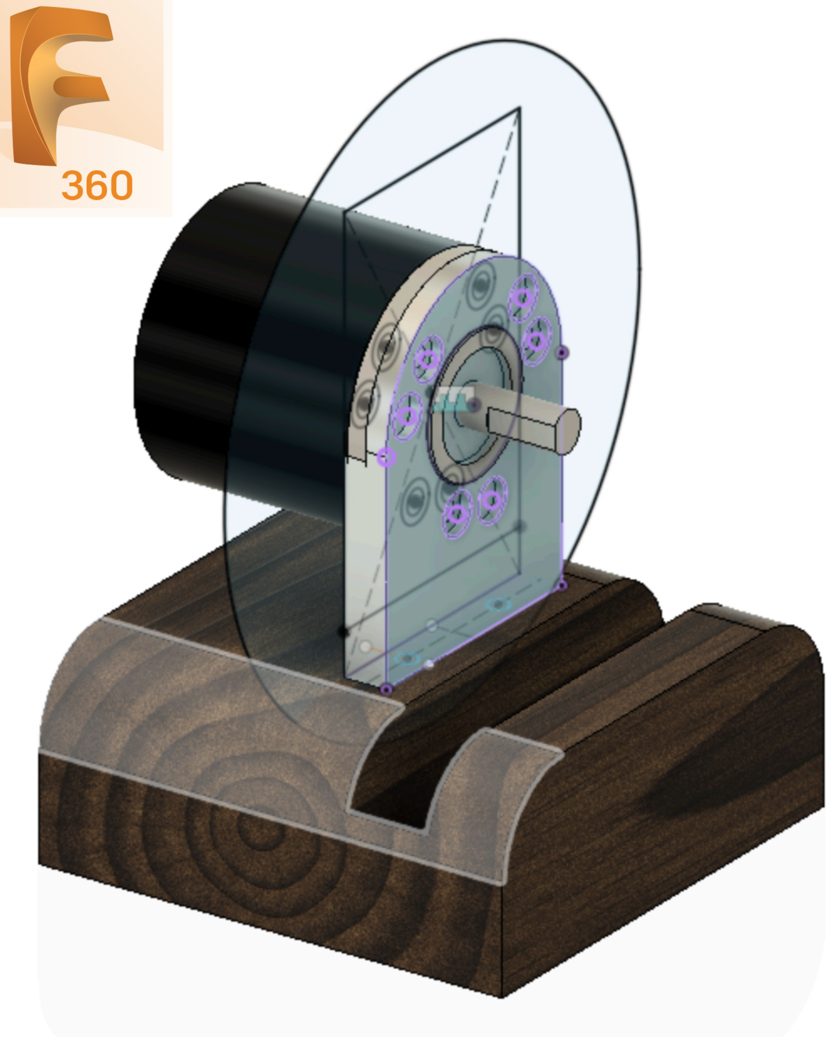
Other ideas?

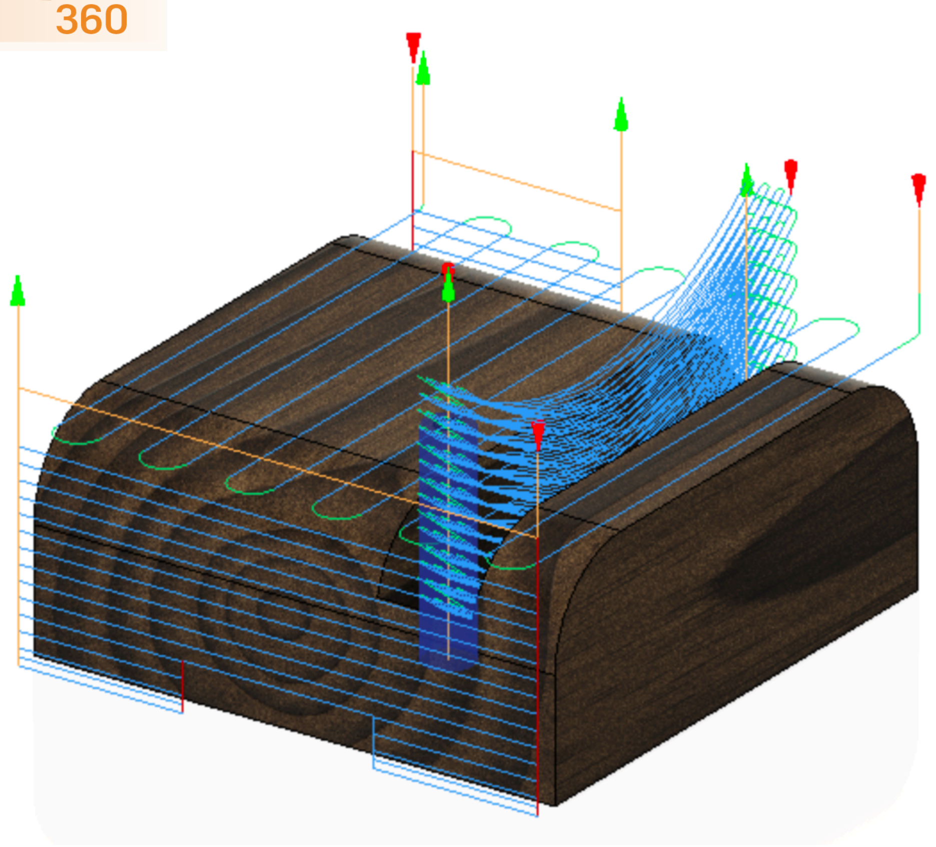


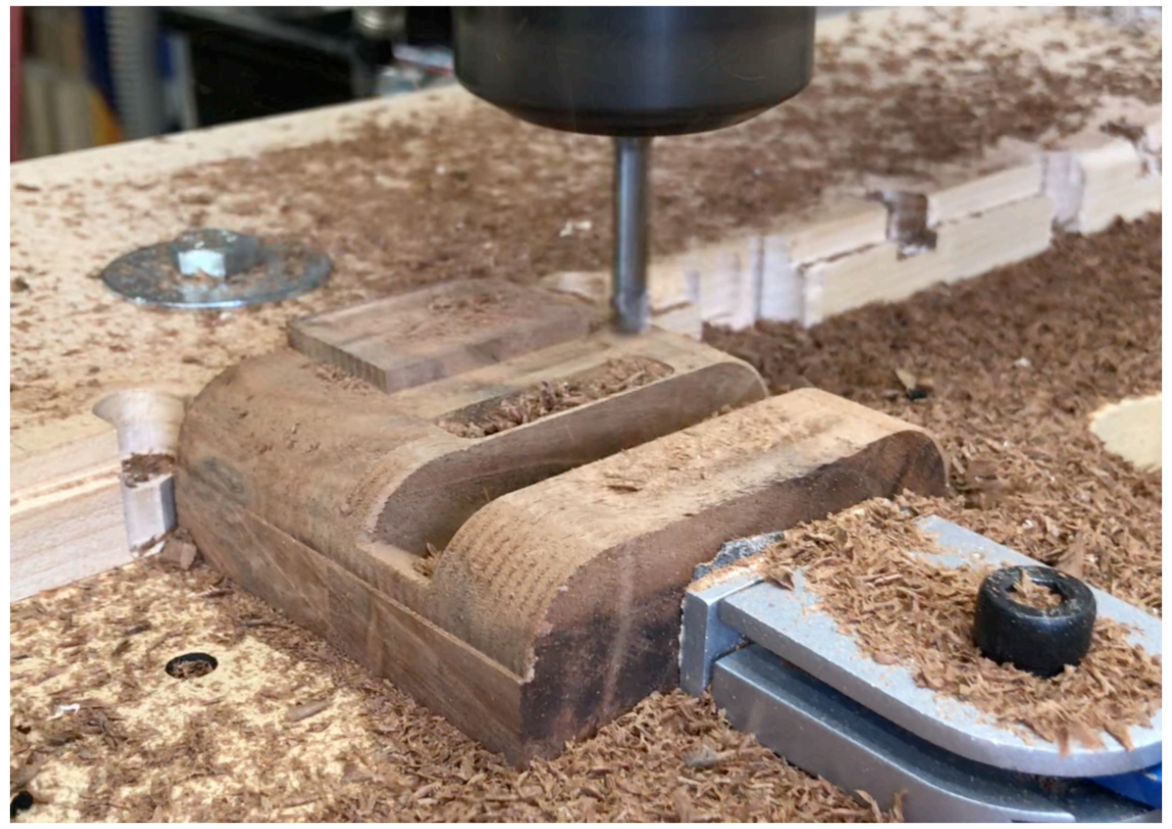
CAD

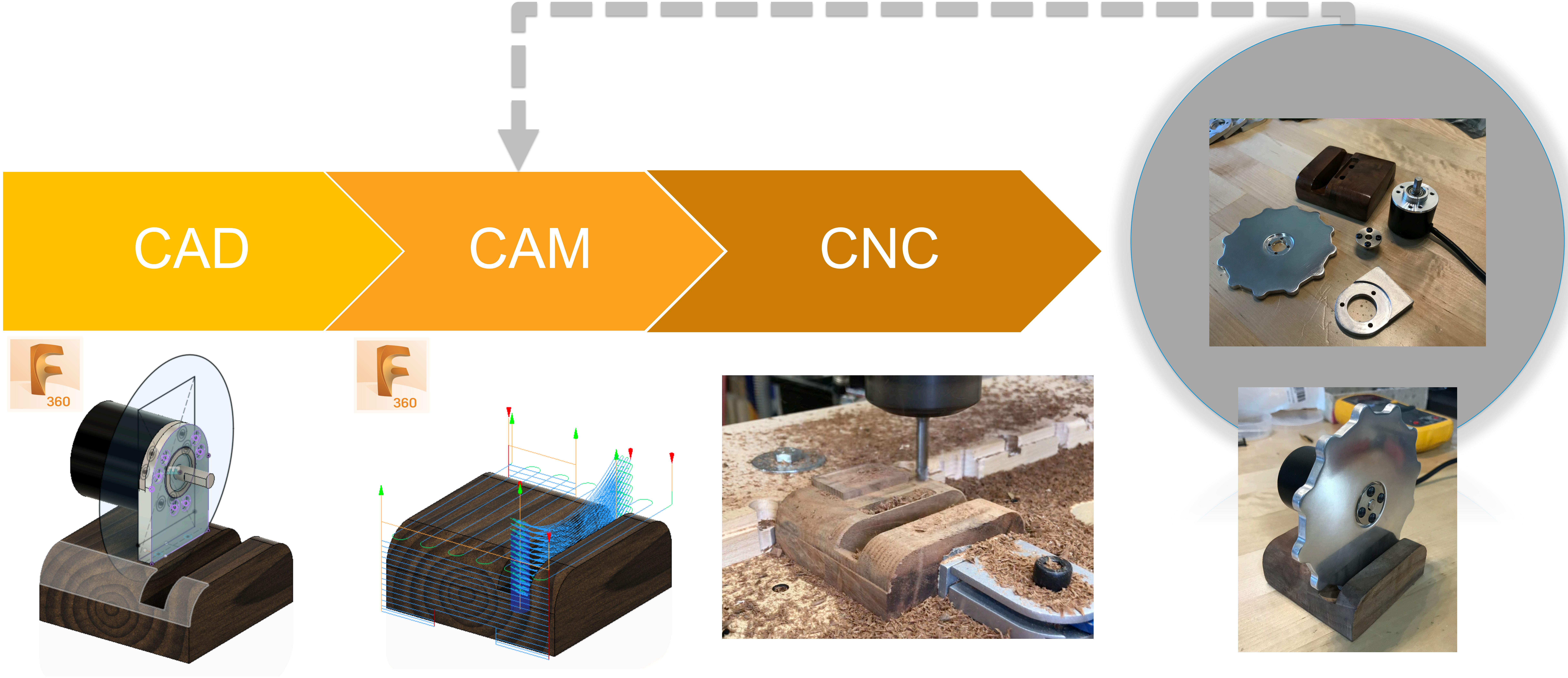
CAM

CNC





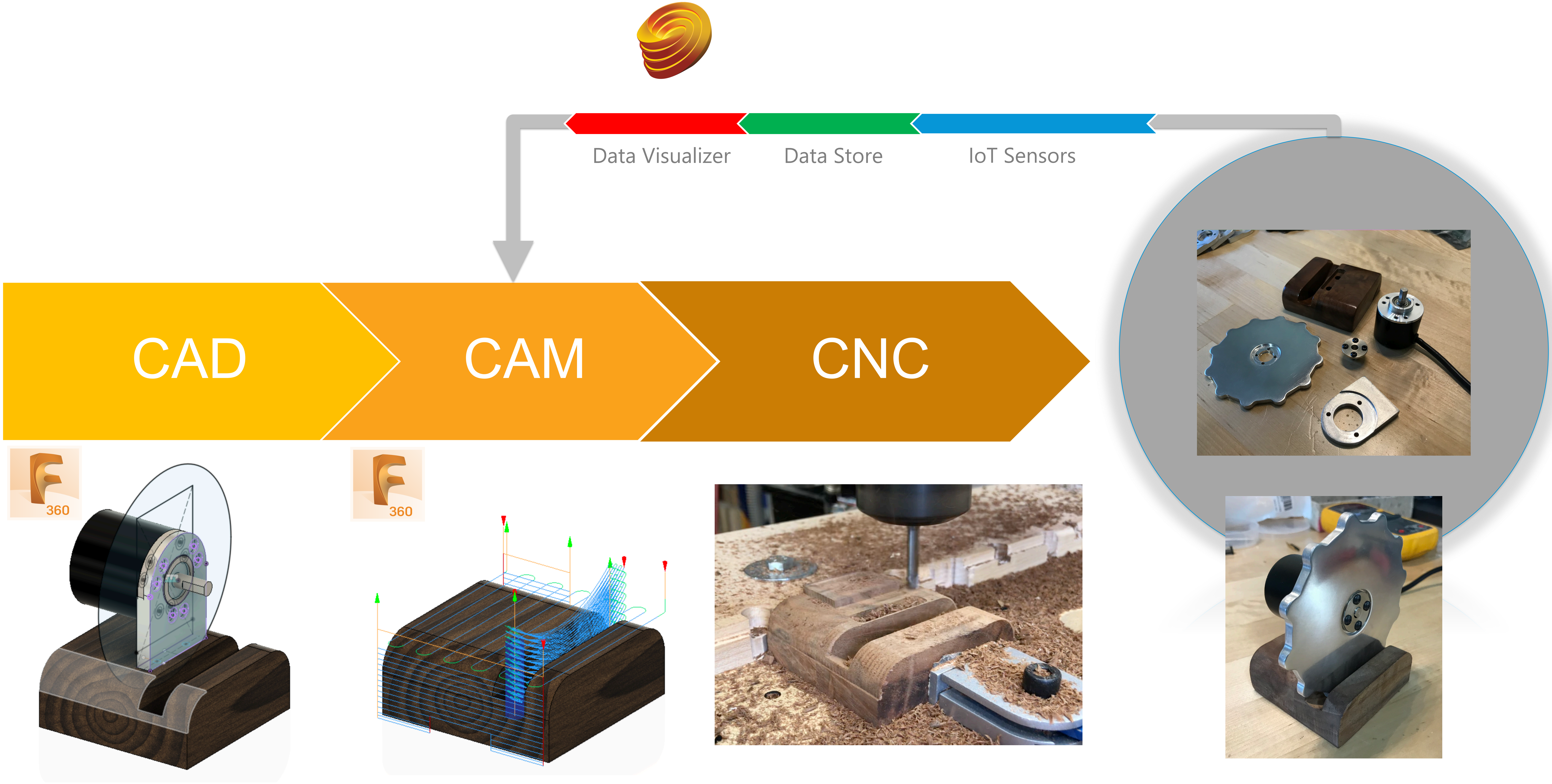




CAD

CAM

CNC



Data Visualizer

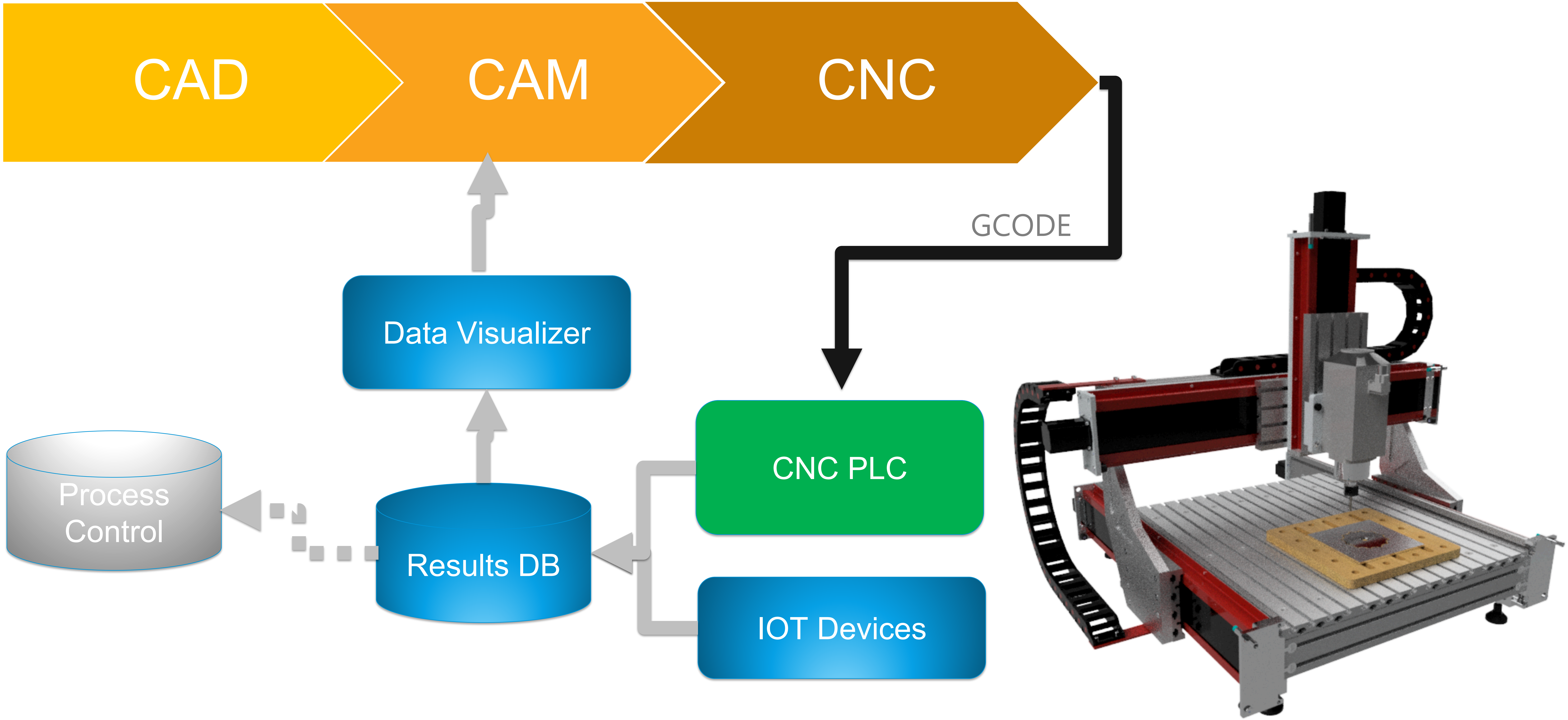
Data Store

IoT Sensors

CAD

CAM

CNC



CAD

CAM

CNC

GCODE

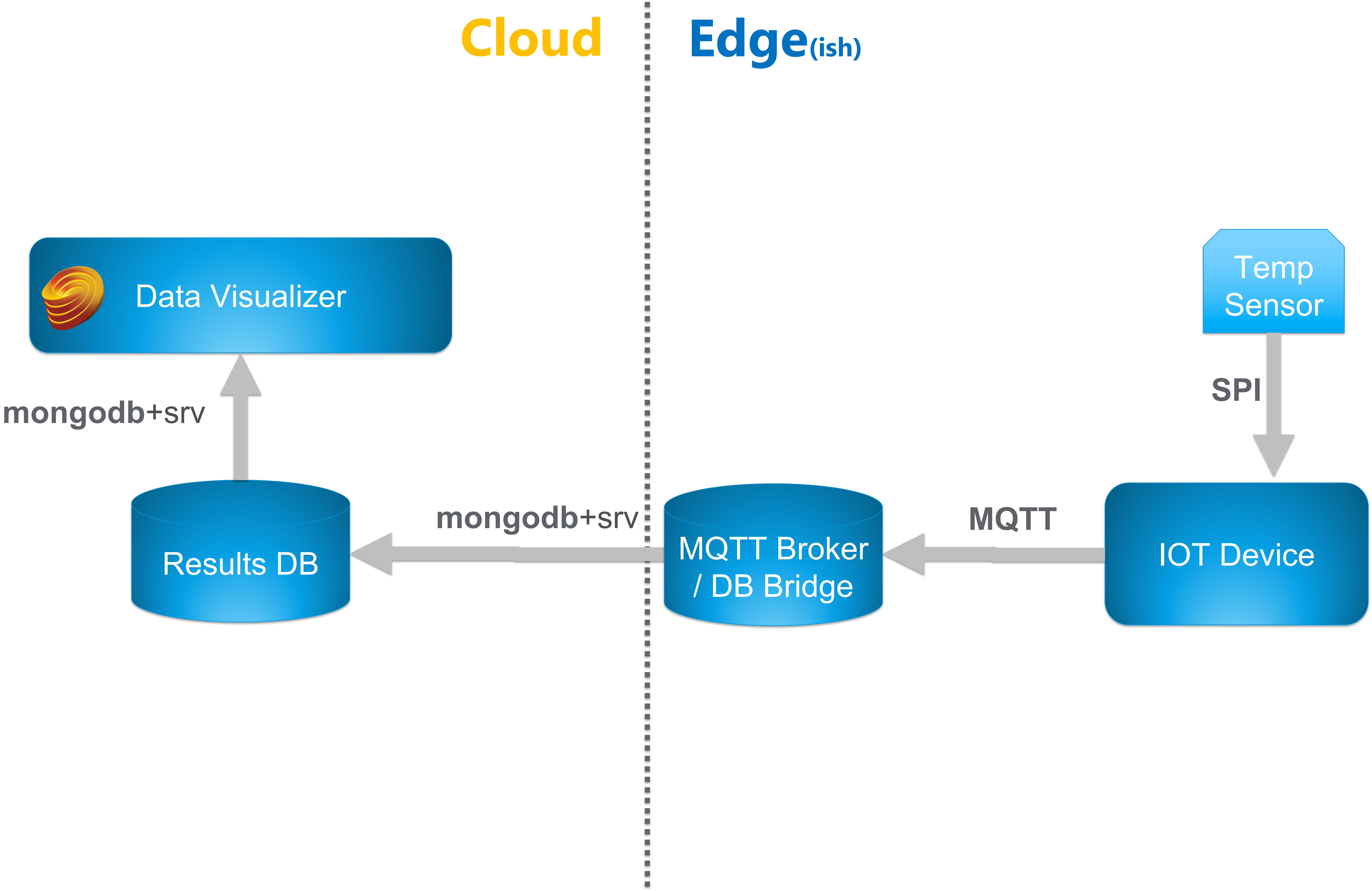
Data Visualizer

CNC PLC

Process Control

Results DB

IOT Devices



Cloud

Edge(ish)

Data Visualizer

mongodb+srv

Results DB

mongodb+srv

MQTT Broker / DB Bridge

MQTT

IOT Device

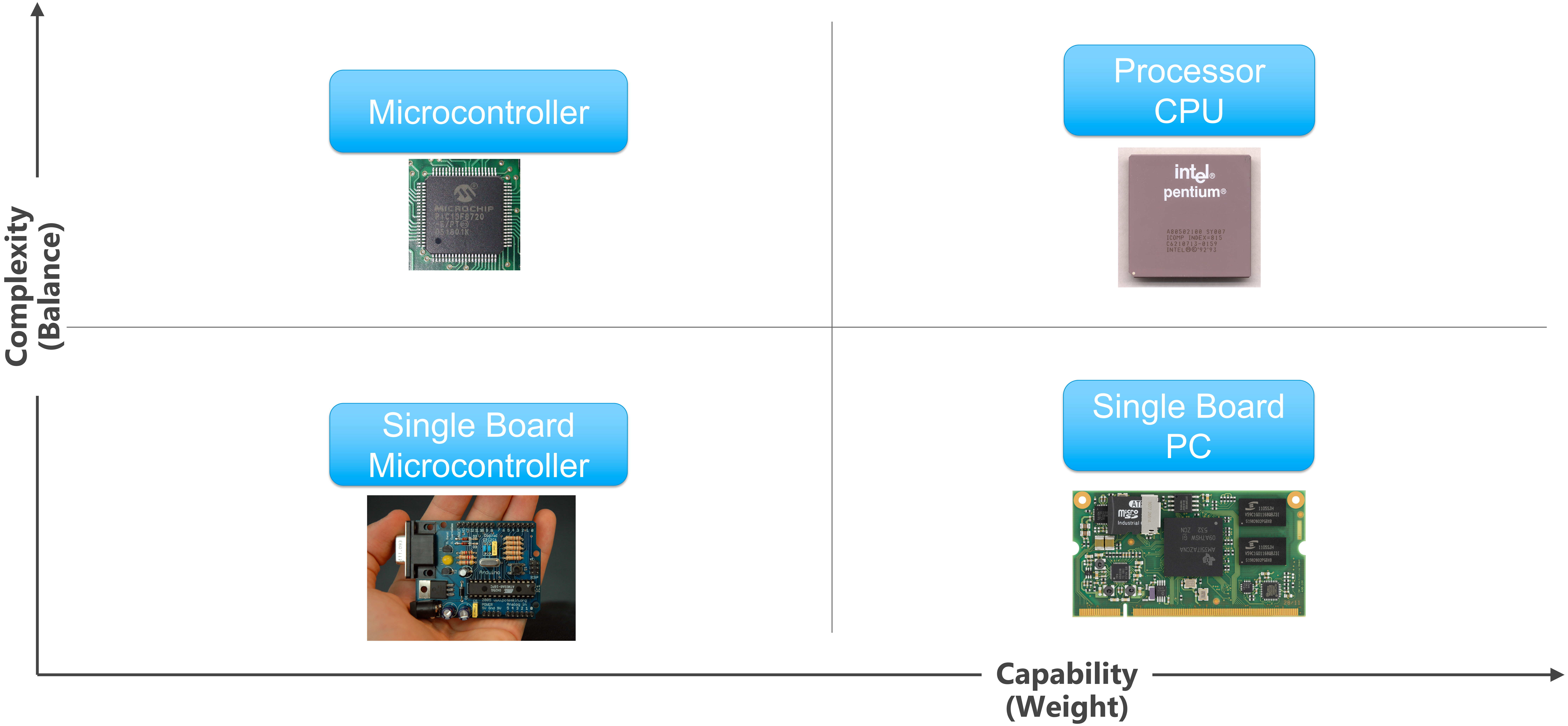
SPI

Temp Sensor

simplify, then **add lightness**

Colin Chapman

Compute Solution Matrix



Microcontroller

Processor CPU

Single Board Microcontroller

Single Board PC

Complexity (Balance)

Capability (Weight)

NodeMCU ESP12E (WiFi with Limited-IO)

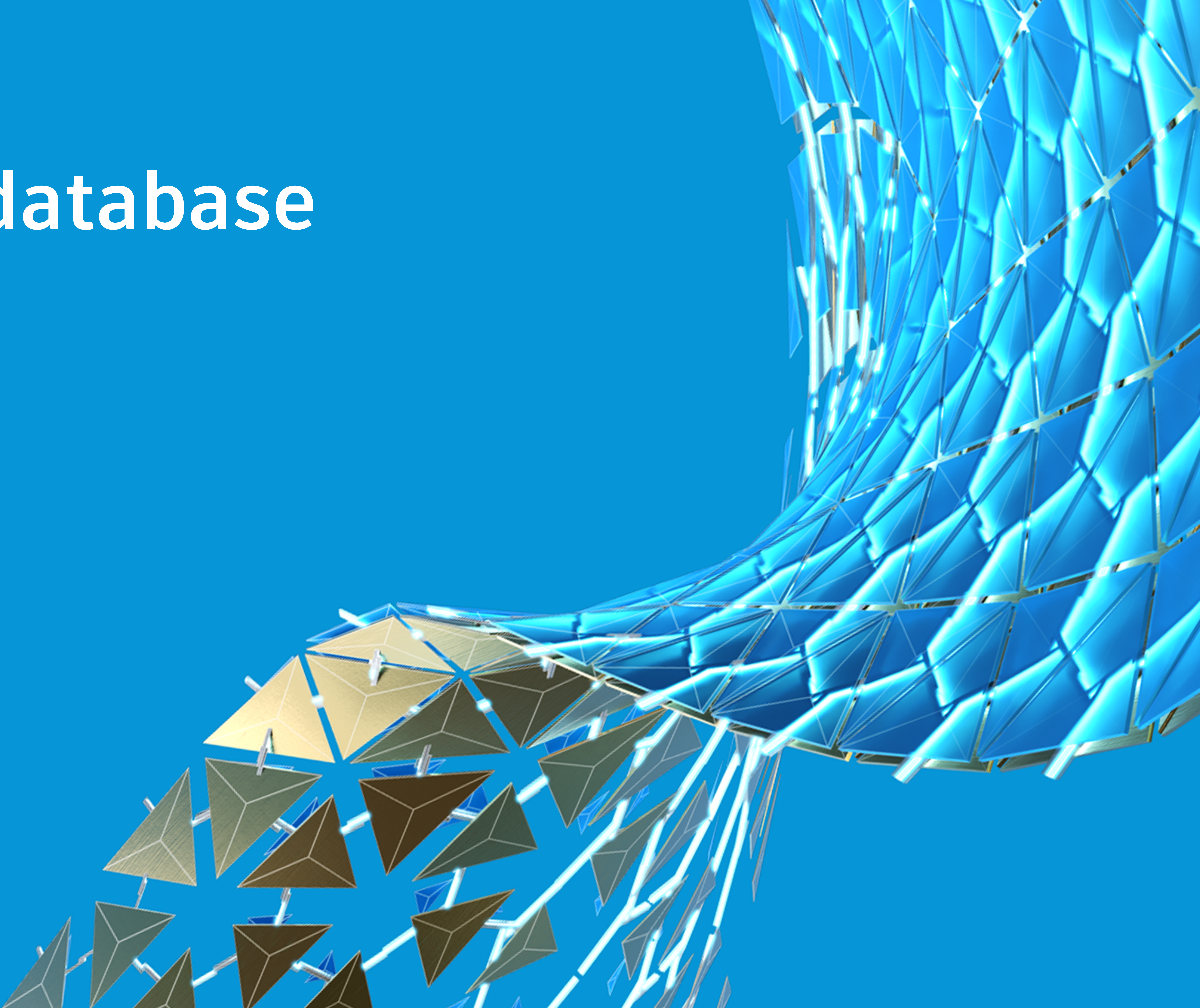
Our Solution Recipe

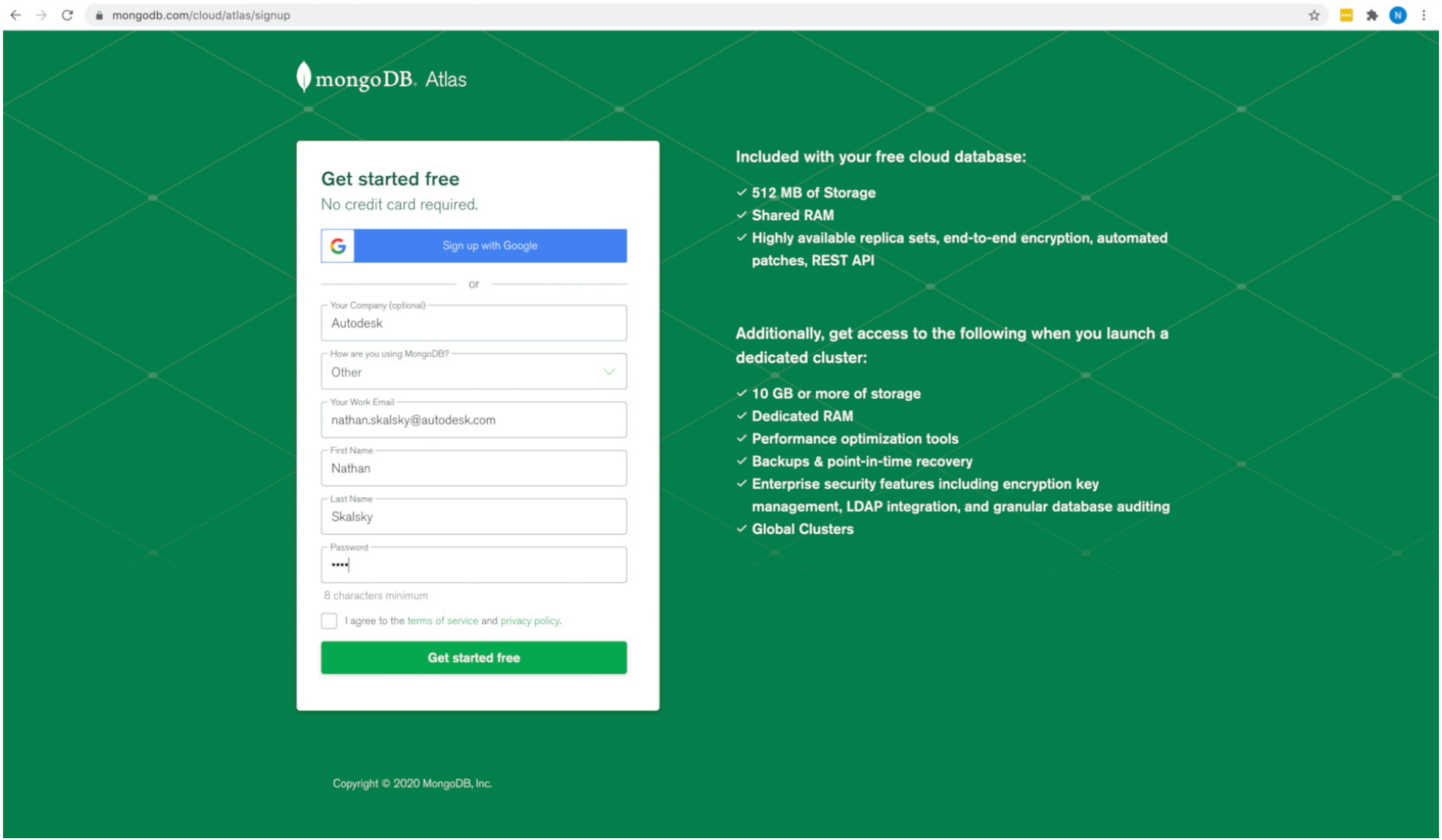
Visualization Engine	**Forge Viewer**
Database	**MongoDB**
DB Protocol	**MongoDB+SRV**
IoT Protocol	**MQTT**
IoT Device	**ESP8266 NodeMCU ESP-12E**
Temperature Sensor	**DS18B20**

Our Plan

- Deploy database
- Build the MQTT broker
- Build the IoT sensor
- Build the IoT firmware
- Test MQTT → Database
- Translate our Fusion 360 model
- Build Forge visualization service
- Put it all Together
- Wrap things up

Deploy the database





mongodb.com/cloud/atlas/signup

mongoDB. Atlas

Get started free

No credit card required.

Sign up with Google

or

Your Company (optional): Autodesk

How are you using MongoDB?: Other

Your Work Email: nathan.skalsky@autodesk.com

First Name: Nathan

Last Name: Skalsky

Password: ••••

8 characters minimum

□ I agree to the terms of service and privacy policy.

Get started free

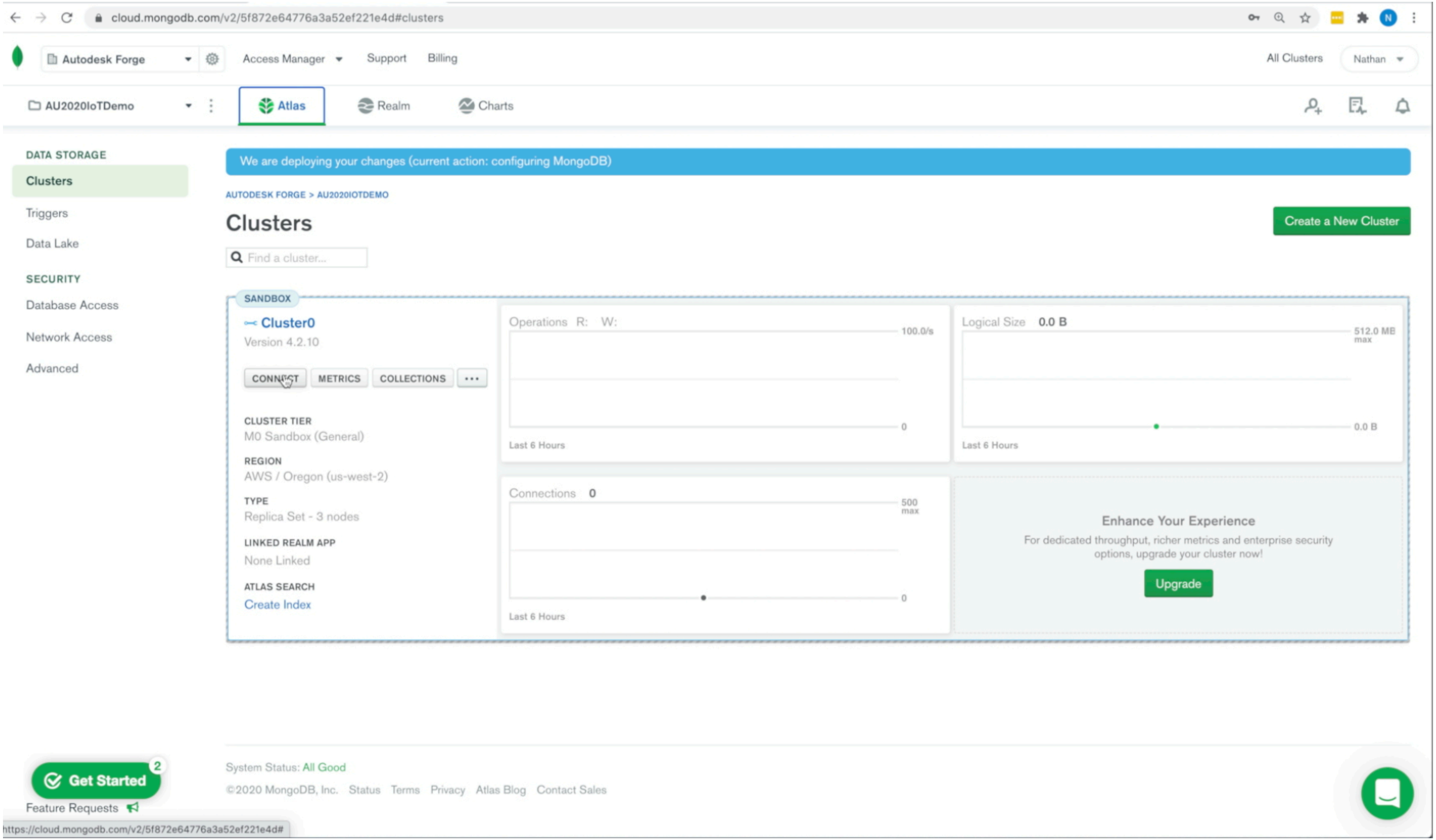
Included with your free cloud database:

- ✓ 512 MB of Storage
- ✓ Shared RAM
- ✓ Highly available replica sets, end-to-end encryption, automated patches, REST API

Additionally, get access to the following when you launch a dedicated cluster:

- ✓ 10 GB or more of storage
- ✓ Dedicated RAM
- ✓ Performance optimization tools
- ✓ Backups & point-in-time recovery
- ✓ Enterprise security features including encryption key management, LDAP integration, and granular database auditing
- ✓ Global Clusters

Copyright © 2020 MongoDB, Inc.



We are deploying your changes (current action: configuring MongoDB)

AUTODESK FORGE > AU2020IOTDEMO

Clusters

Cluster0
Version 4.2.10

CLUSTER TIER
M0 Sandbox (General)

REGION
AWS / Oregon (us-west-2)

TYPE
Replica Set - 3 nodes

LINKED REALM APP
None Linked

ATLAS SEARCH
Create Index

Operations R: W:
100.0/s
0
Last 6 Hours

Logical Size 0.0 B
512.0 MB max
0.0 B
Last 6 Hours

Connections 0
500 max
0
Last 6 Hours

Enhance Your Experience

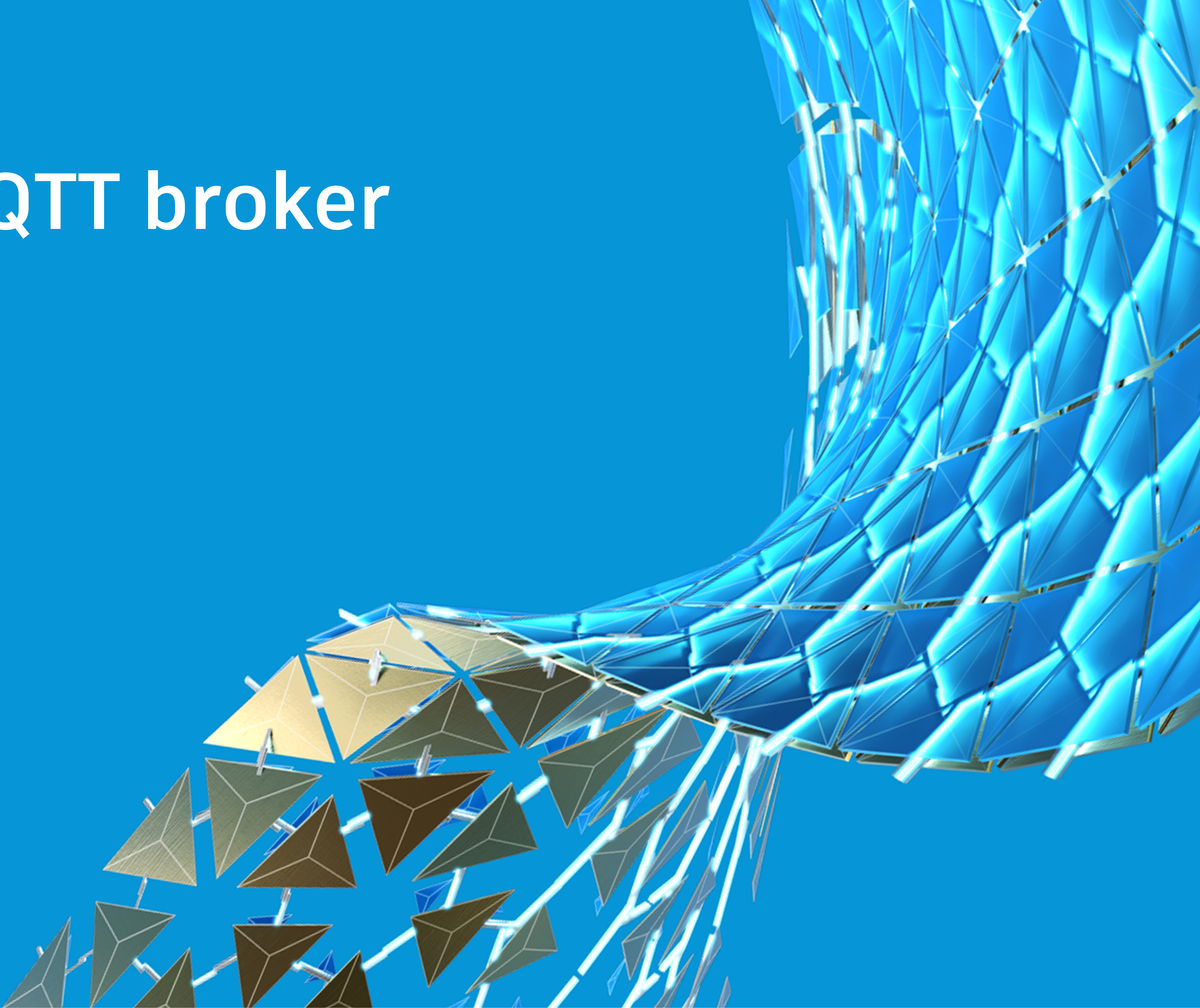
For dedicated throughput, richer metrics and enterprise security options, upgrade your cluster now!

System Status: All Good

©2020 MongoDB, Inc. Status Terms Privacy Atlas Blog Contact Sales

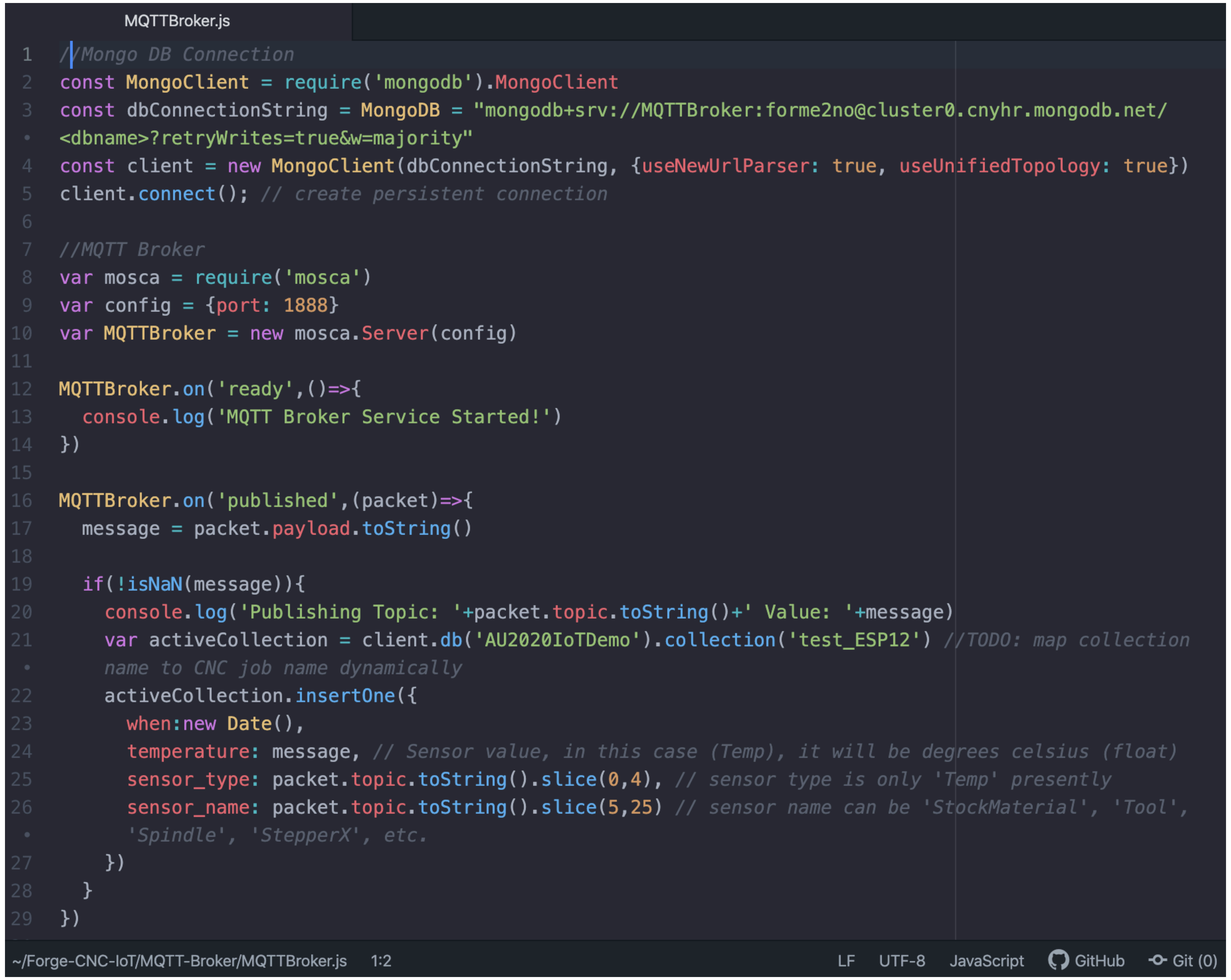
https://cloud.mongodb.com/v2/5f872e64776a3a52ef221e4d#

Build the MQTT broker



Building a Simple MQTT Broker (+MongoDB Bridge)

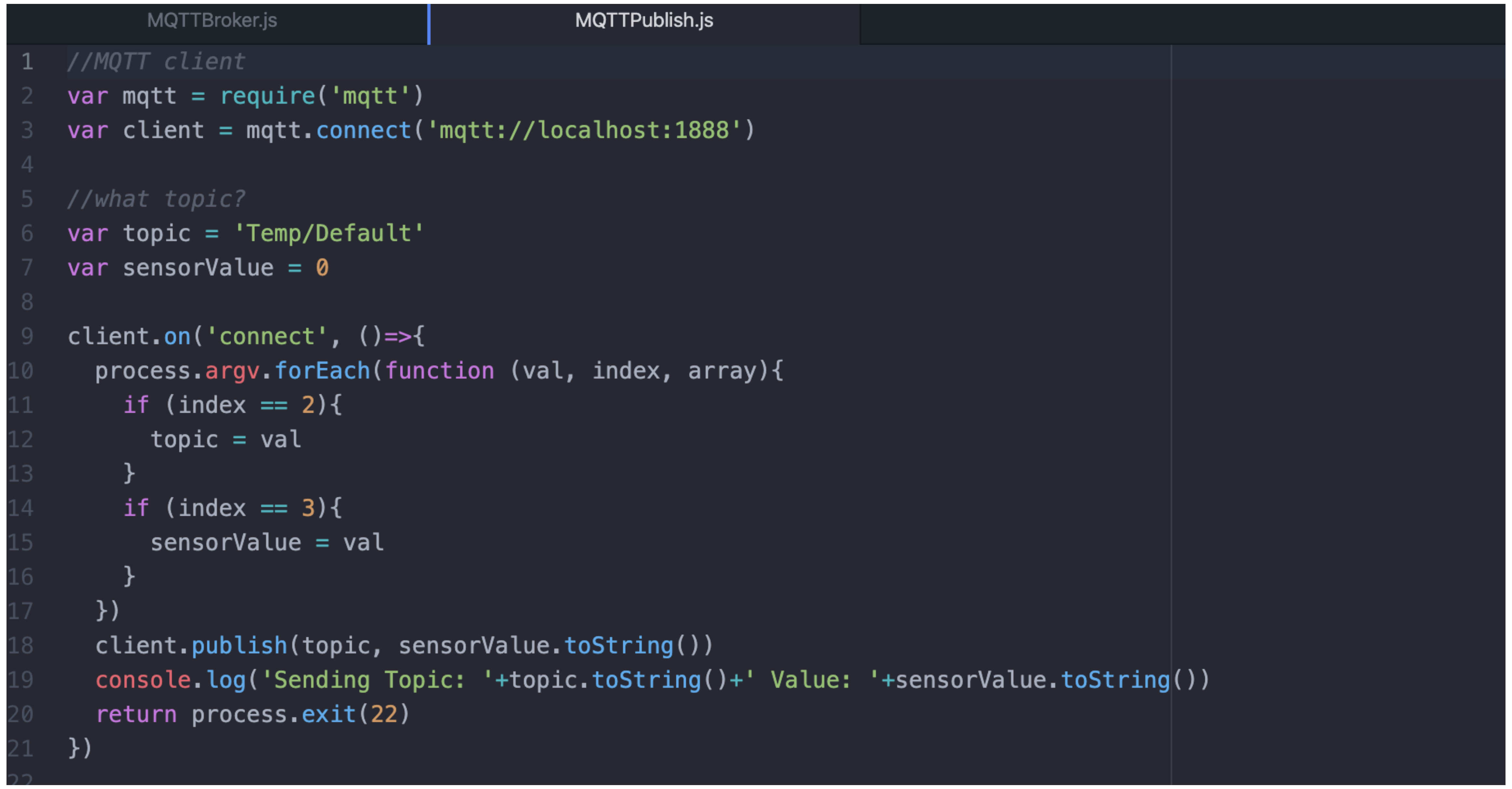
- **Libraries Used:** mongodb, mosca (MQTT)
- **Description:** Listens to port 1888, receives MQTT messages, inserts them into the results database using a simple sensor schema



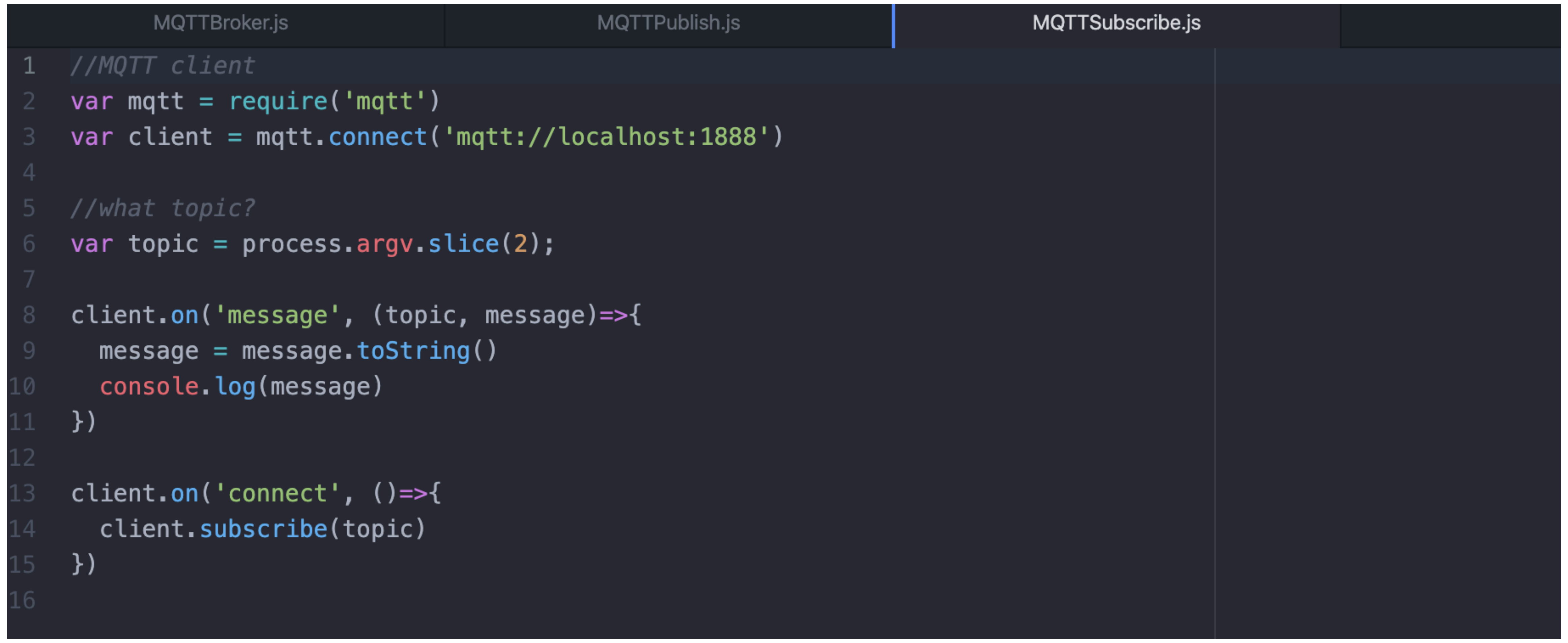
```javascript
//Mongo DB Connection
const MongoClient = require('mongodb').MongoClient
const dbConnectionString = MongoDB = "mongodb+srv://MQTTBroker:forme2no@cluster0.cnyhr.mongodb.net/
<dbname>?retryWrites=true&w=majority"
const client = new MongoClient(dbConnectionString, {useNewUrlParser: true, useUnifiedTopology: true})
client.connect(); // create persistent connection

//MQTT Broker
var mosca = require('mosca')
var config = {port: 1888}
var MQTTBroker = new mosca.Server(config)

MQTTBroker.on('ready',()=>{
  console.log('MQTT Broker Service Started!')
})

MQTTBroker.on('published',(packet)=>{
  message = packet.payload.toString()

  if(!isNaN(message)){
    console.log('Publishing Topic: '+packet.topic.toString()+' Value: '+message)
    var activeCollection = client.db('AU2020IoTDemo').collection('test_ESP12') //TODO: map collection
    name to CNC job name dynamically
    activeCollection.insertOne({
      when:new Date(),
      temperature: message, // Sensor value, in this case (Temp), it will be degrees celsius (float)
      sensor_type: packet.topic.toString().slice(0,4), // sensor type is only 'Temp' presently
      sensor_name: packet.topic.toString().slice(5,25) // sensor name can be 'StockMaterial', 'Tool',
      'Spindle', 'StepperX', etc.
    })
  }
})
```

Testing a Simple MQTT Broker

- **Libraries Used:** mongodb, mosca (MQTT)
- **Description:**
  - **MQTTPublish.js** – Will send MQTT messages when the user passes in Topic/subtopic string and the sensor value
  - **MQTTSubscribe.js** – Will subscribe to receive MQTT messages using the specified topic/subtopic string.

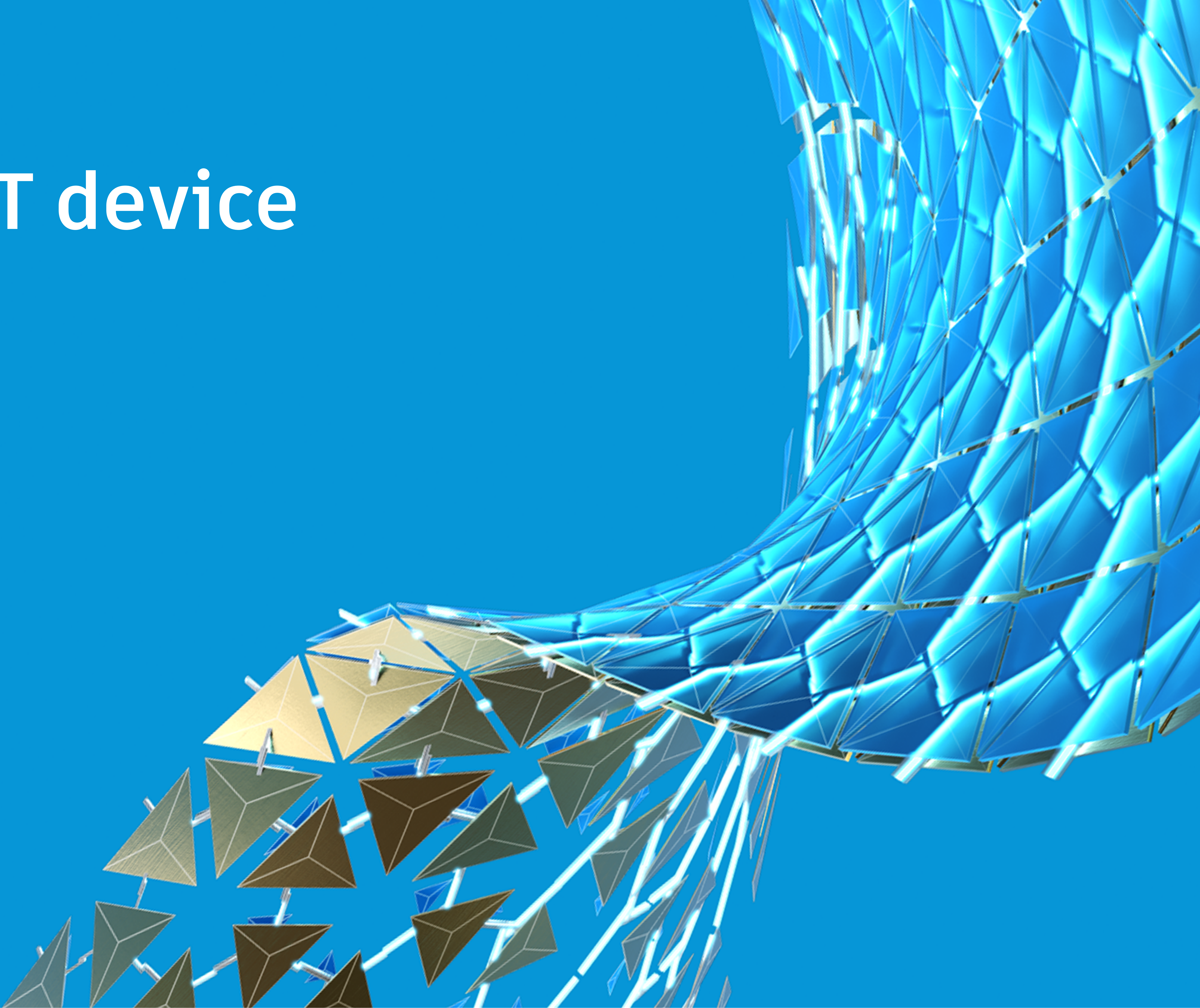


```javascript
//MQTT client
var mqtt = require('mqtt')
var client = mqtt.connect('mqtt://localhost:1888')

//what topic?
var topic = 'Temp/Default'
var sensorValue = 0

client.on('connect', ()=>{
  process.argv.forEach(function (val, index, array){
    if (index == 2){
      topic = val
    }
    if (index == 3){
      sensorValue = val
    }
  })
  client.publish(topic, sensorValue.toString())
  console.log('Sending Topic: '+topic.toString()+' Value: '+sensorValue.toString())
  return process.exit(22)
})
```



```javascript
//MQTT client
var mqtt = require('mqtt')
var client = mqtt.connect('mqtt://localhost:1888')

//what topic?
var topic = process.argv.slice(2);

client.on('message', (topic, message)=>{
  message = message.toString()
  console.log(message)
})

client.on('connect', ()=>{
  client.subscribe(topic)
})
```

Build the IoT device



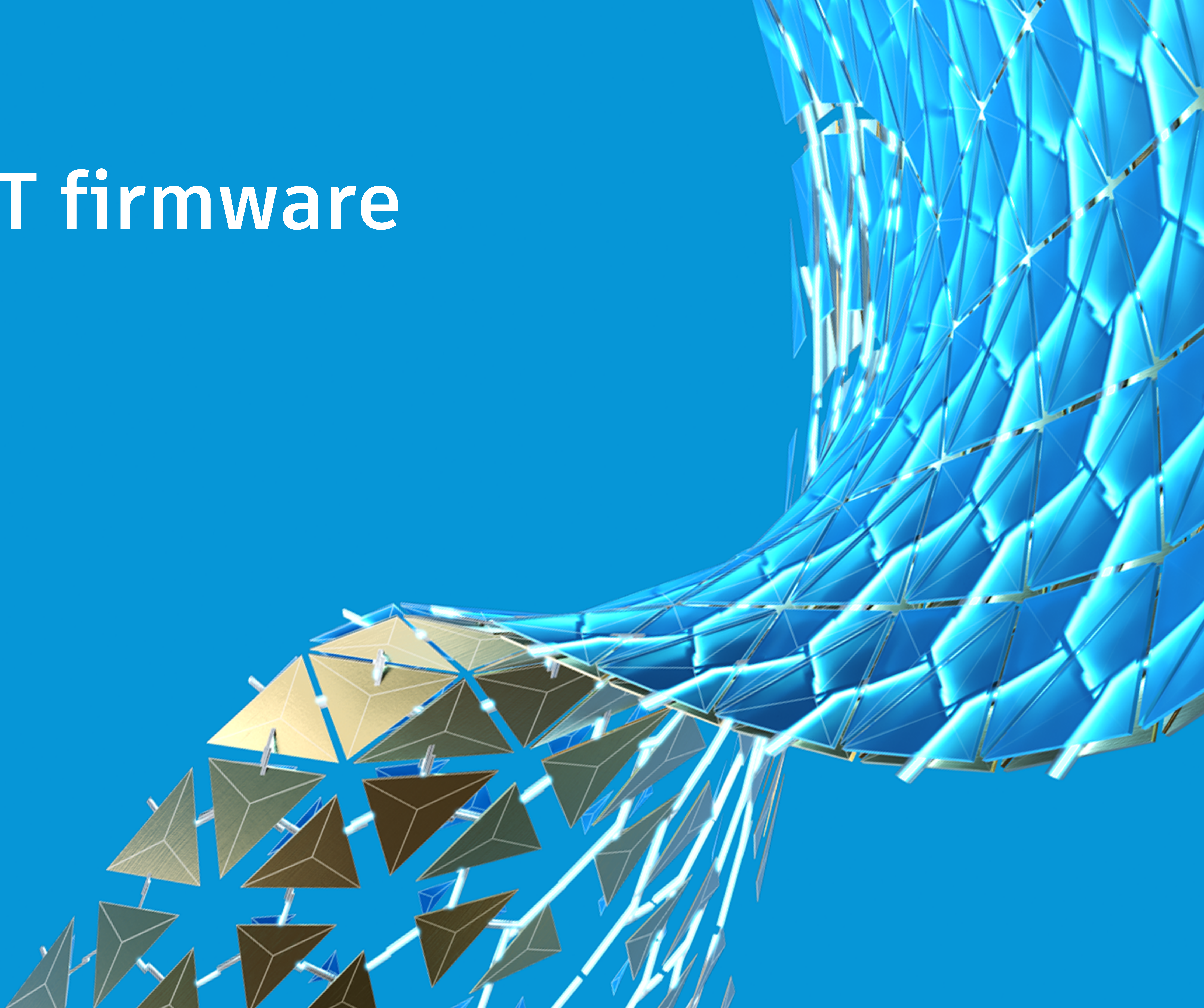
Building IoT device

- **See class video and handout for details**

- **BOM**:
  - ESP8266 NodeMCU ESP-12E
  - DS18B20 Digital Thermometer
  - 4.7k Resister

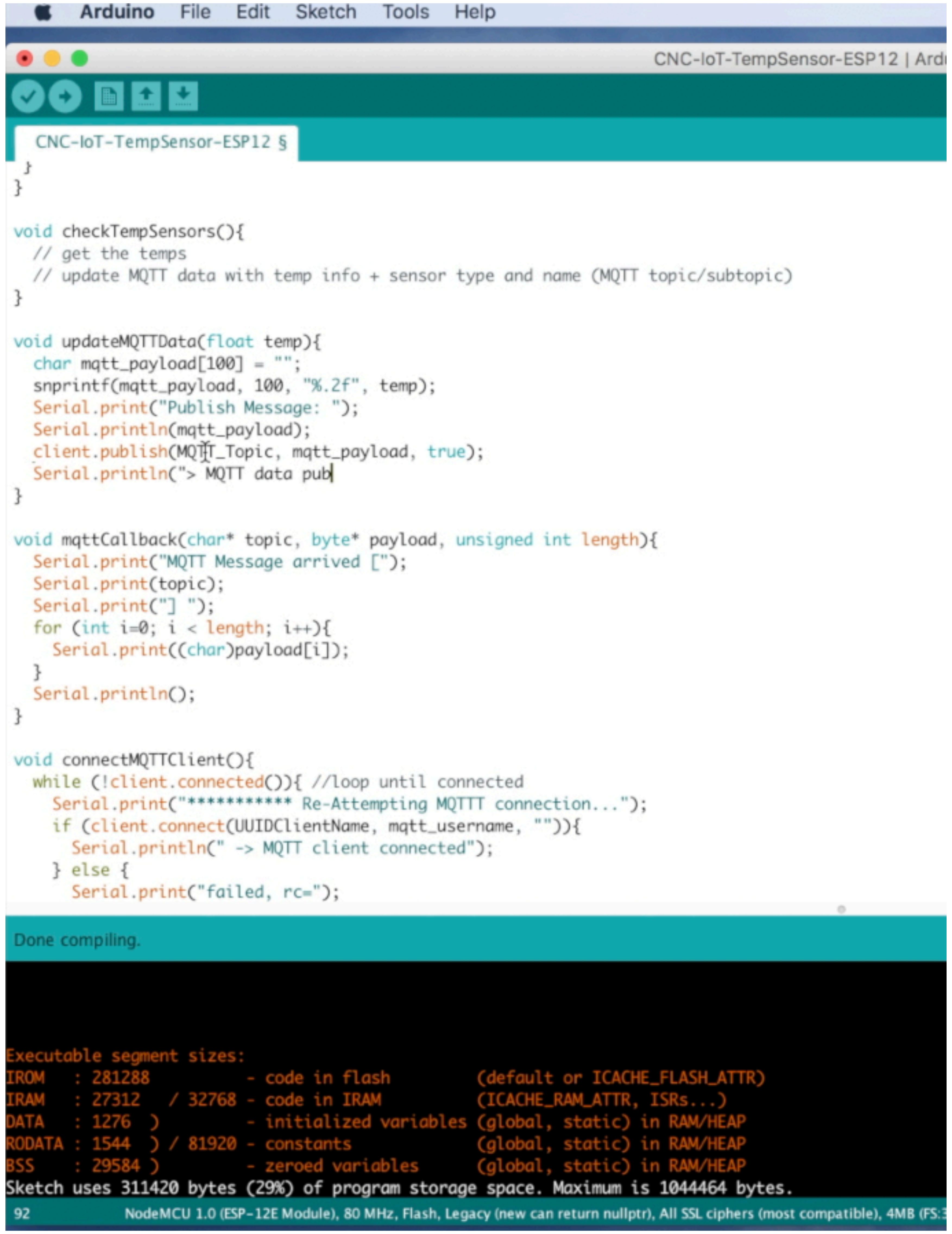
- **Connections**:
  - Temp Sensor.VCC = Vin on ESP board
  - Temp Sensor.GND = GND on ESP Board
  - Temp Sensor.Data = GPIO 5
  - 4.7k resister connected to GPIO 5 and Vin

Build the IoT firmware



Build the IoT Firmware

- **See class video and handout for details**
- **Nathan's Four Essential IoT Device Features**
  1. WiFi client to connect to access point
  2. Ability to act as WiFi access point to receive WiFi credentials from user
  3. Feature to allow user to reset controller to an unconfigured state (double reset)
  4. MQTT Client to publish and subscribe to MQTT events



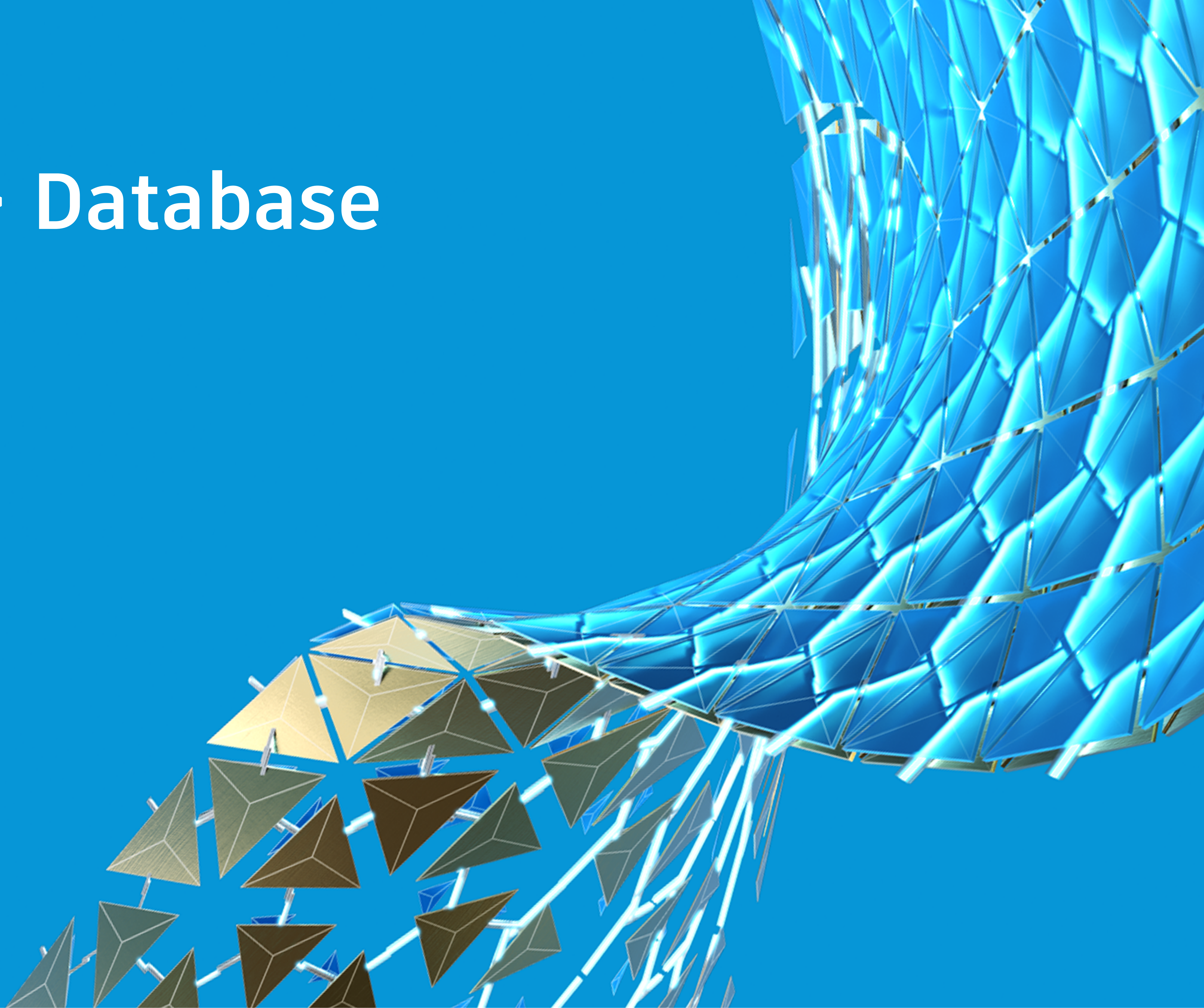
```
  }
}

void checkTempSensors(){
  // get the temps
  // update MQTT data with temp info + sensor type and name (MQTT topic/subtopic)
}

void updateMQTTData(float temp){
  char mqtt_payload[100] = "";
  snprintf(mqtt_payload, 100, "%.2f", temp);
  Serial.print("Publish Message: ");
  Serial.println(mqtt_payload);
  client.publish(MQTT_Topic, mqtt_payload, true);
  Serial.println("> MQTT data pub
}

void mqttCallback(char* topic, byte* payload, unsigned int length){
  Serial.print("MQTT Message arrived [");
  Serial.print(topic);
  Serial.print("] ");
  for (int i=0; i < length; i++){
    Serial.print((char)payload[i]);
  }
  Serial.println();
}

void connectMQTTClient(){
  while (!client.connected()){ //loop until connected
    Serial.print("*********** Re-Attempting MQTTT connection...");
    if (client.connect(UUIDClientName, mqtt_username, "")){
      Serial.println(" -> MQTT client connected");
    } else {
      Serial.print("failed, rc=");
```

Done compiling.

```
Executable segment sizes:
IROM   : 281288          - code in flash         (default or ICACHE_FLASH_ATTR)
IRAM   : 27312   / 32768 - code in IRAM          (ICACHE_RAM_ATTR, ISRs...)
DATA   : 1276  )         - initialized variables (global, static) in RAM/HEAP
RODATA : 1544  ) / 81920 - constants             (global, static) in RAM/HEAP
BSS    : 29584 )         - zeroed variables      (global, static) in RAM/HEAP
Sketch uses 311420 bytes (29%) of program storage space. Maximum is 1044464 bytes.
```

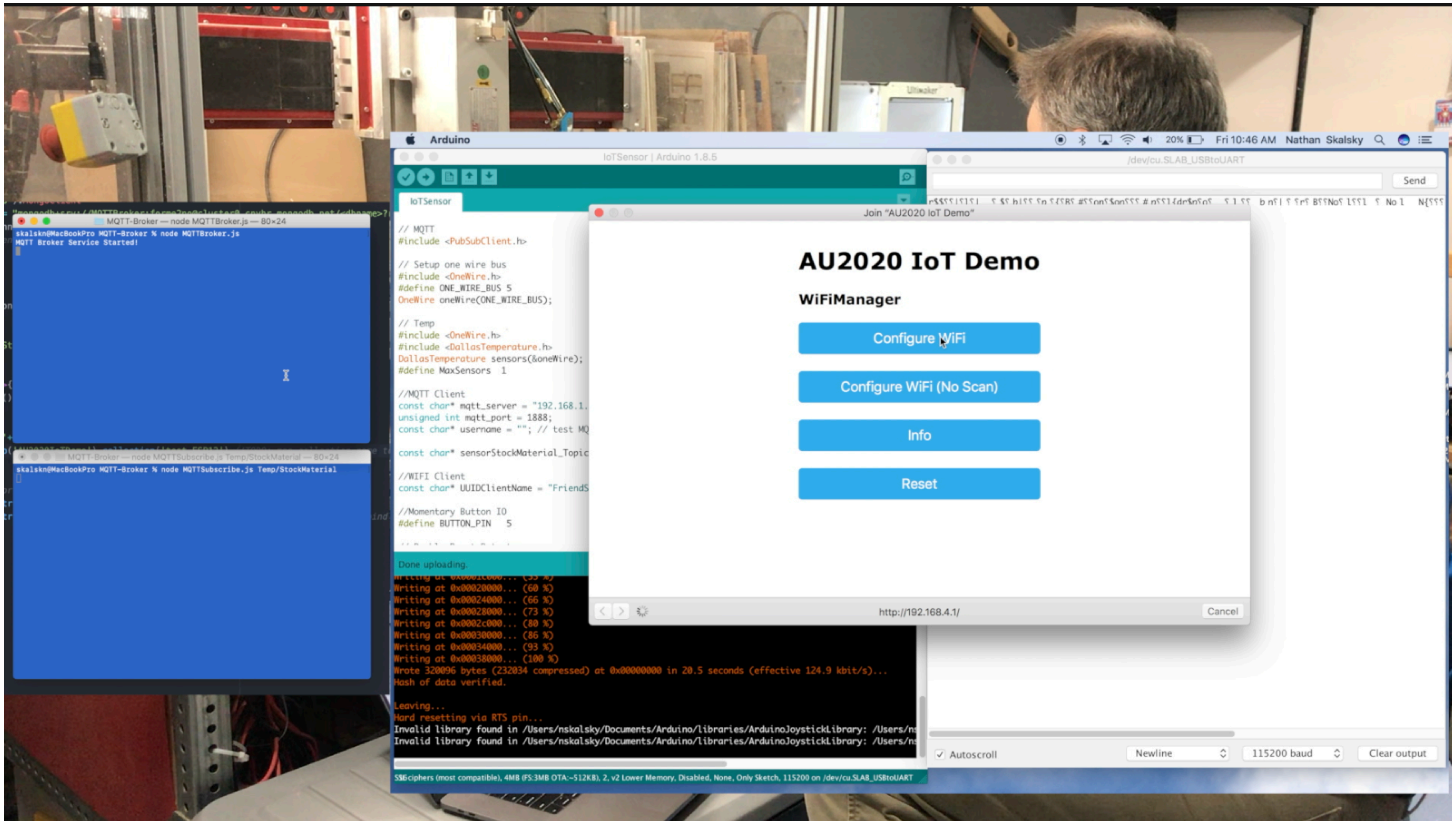
92 NodeMCU 1.0 (ESP-12E Module), 80 MHz, Flash, Legacy (new can return nullptr), All SSL ciphers (most compatible), 4MB (FS:

Test MQTT > Database



Unit testing MQTT to database

- See class video and handout for details



```
skalskn@MacBookPro MQTT-Broker % node MQTTBroker.js
MQTT Broker Service Started!
```

```
skalskn@MacBookPro MQTT-Broker % node MQTTSubscribe.js Temp/StockMaterial
```

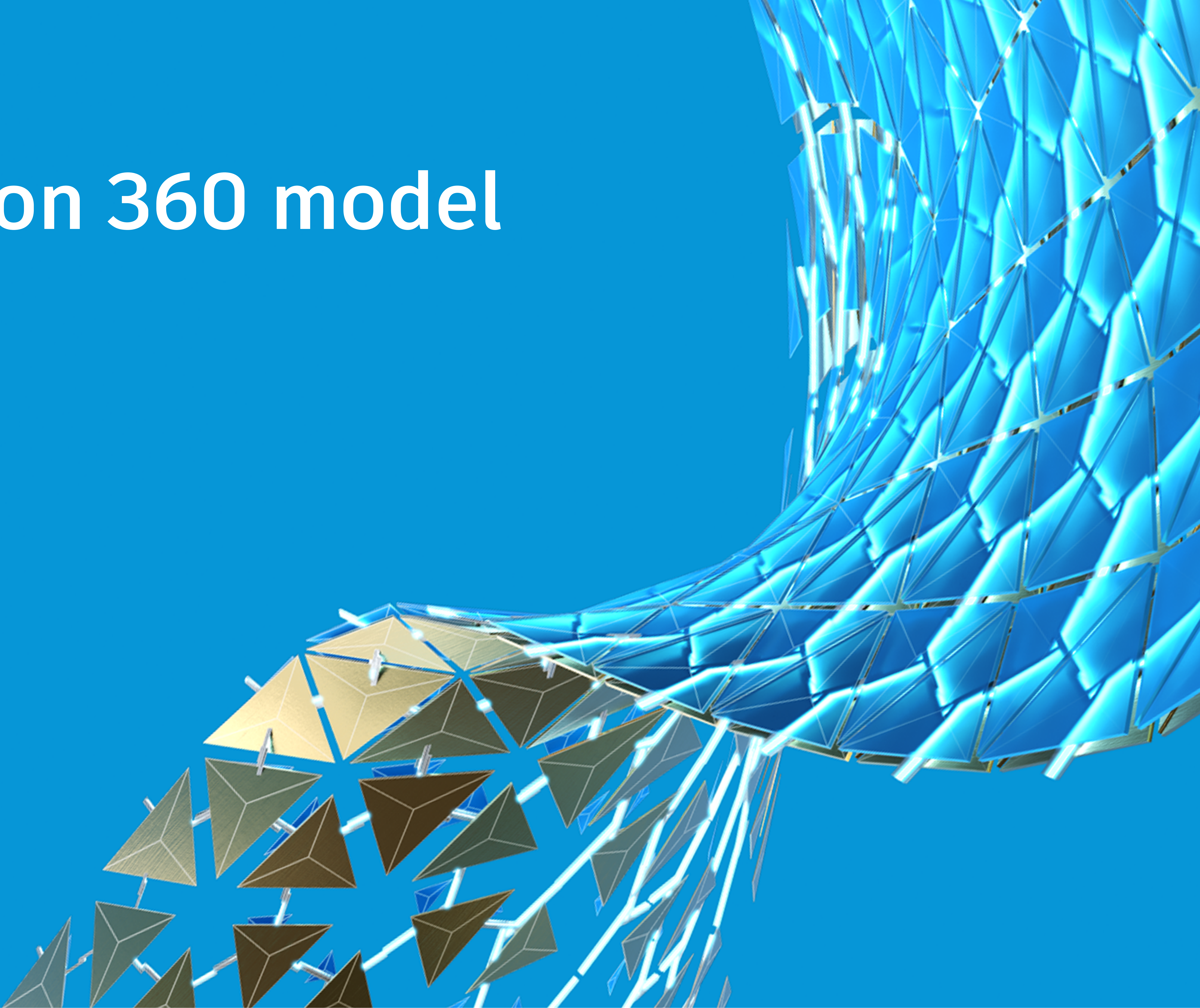
```
// MQTT
#include <PubSubClient.h>

// Setup one wire bus
#include <OneWire.h>
#define ONE_WIRE_BUS 5
OneWire oneWire(ONE_WIRE_BUS);

// Temp
#include <OneWire.h>
#include <DallasTemperature.h>
DallasTemperature sensors(&oneWire);
#define MaxSensors  1

//MQTT Client
const char* mqtt_server = "192.168.1.
unsigned int mqtt_port = 1888;
const char* username = ""; // test MQ

const char* sensorStockMaterial_Topic

//WIFI Client
const char* UUIDClientName = "FriendS

//Momentary Button IO
#define BUTTON_PIN   5
```

Done uploading.

```
Writing at 0x00020000... (60 %)
Writing at 0x00024000... (66 %)
Writing at 0x00028000... (73 %)
Writing at 0x0002c000... (80 %)
Writing at 0x00030000... (86 %)
Writing at 0x00034000... (93 %)
Writing at 0x00038000... (100 %)
Wrote 320096 bytes (232034 compressed) at 0x00000000 in 20.5 seconds (effective 124.9 kbit/s)...
Hash of data verified.

Leaving...
Hard resetting via RTS pin...
Invalid library found in /Users/nskalsky/Documents/Arduino/libraries/ArduinoJoystickLibrary: /Users/ns
Invalid library found in /Users/nskalsky/Documents/Arduino/libraries/ArduinoJoystickLibrary: /Users/ns
```

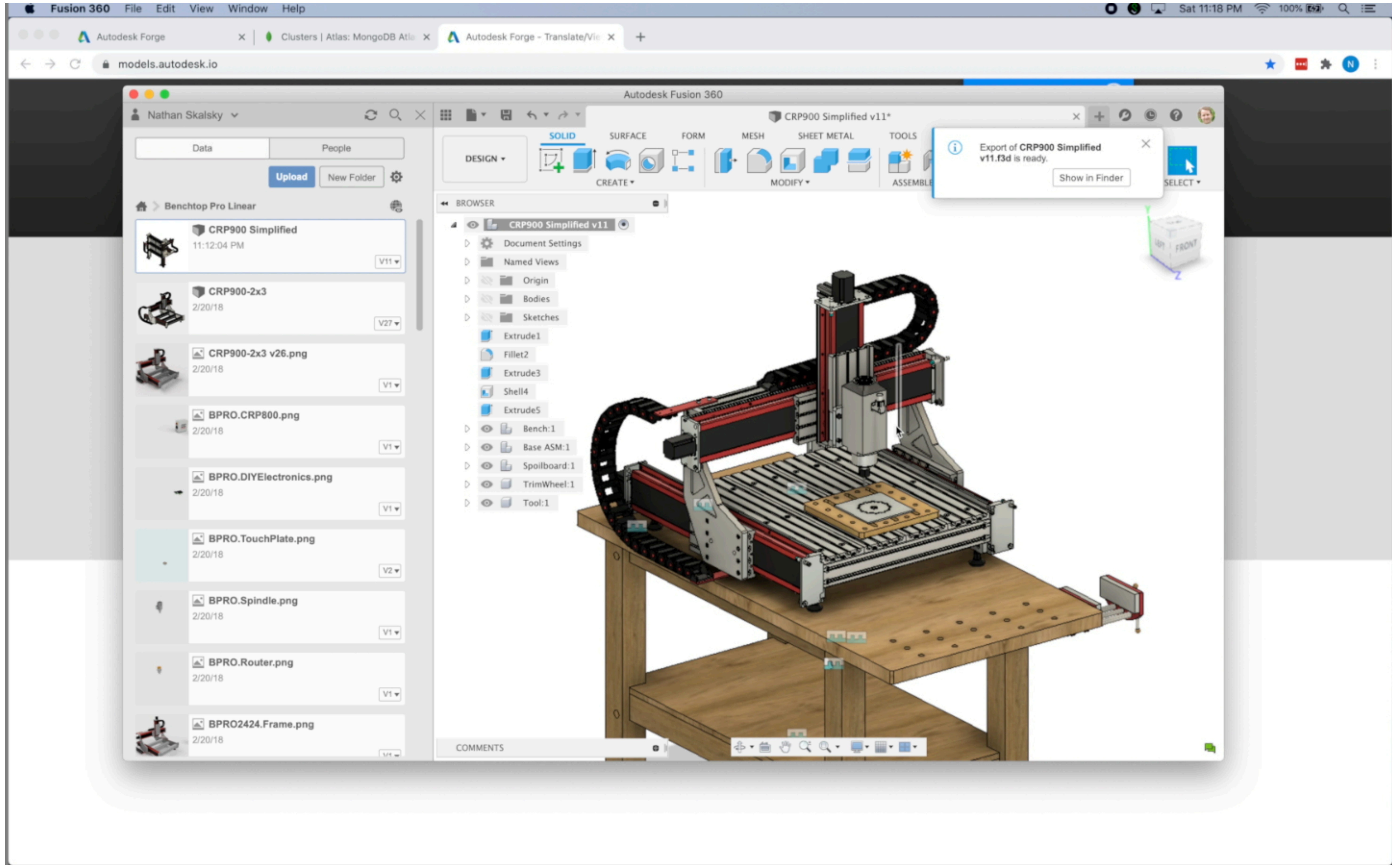
SSLciphers (most compatible), 4MB (FS:3MB OTA:~512KB), 2, v2 Lower Memory, Disabled, None, Only Sketch, 115200 on /dev/cu.SLAB_USBtoUART

Upload Fusion 360 model



Upload Fusion 360 model to Forge

- See class video and handout for details
- Upload model to models.autodesk.io using forge API credentials



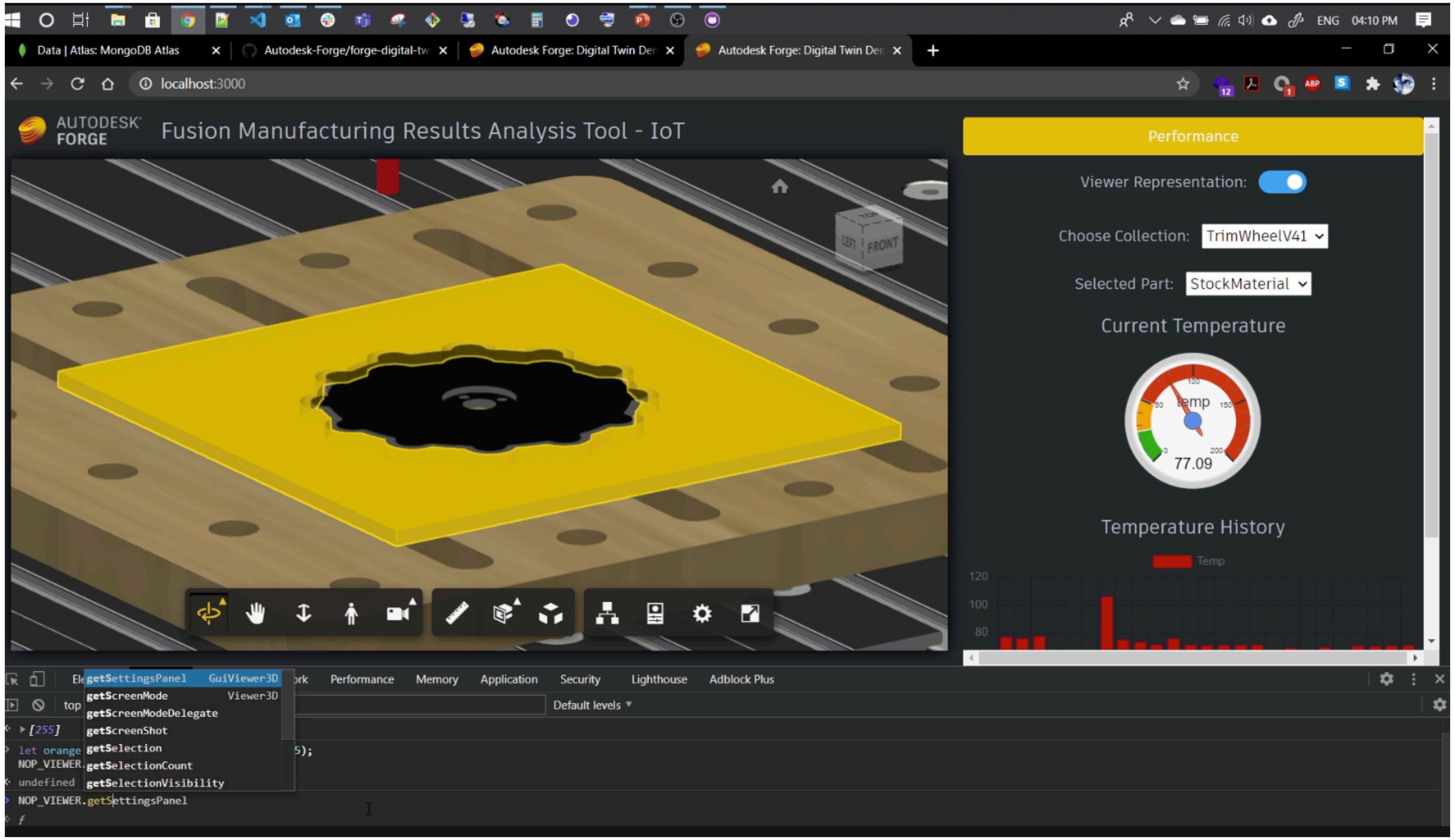
Export of CRP900 Simplified v11.f3d is ready.

Show in Finder

Build the Visualization Service

Build the visualization web experience

- See class video and handout for details



Fusion Manufacturing Results Analysis Tool - IoT

Viewer Representation: ON

Choose Collection: TrimWheelV41

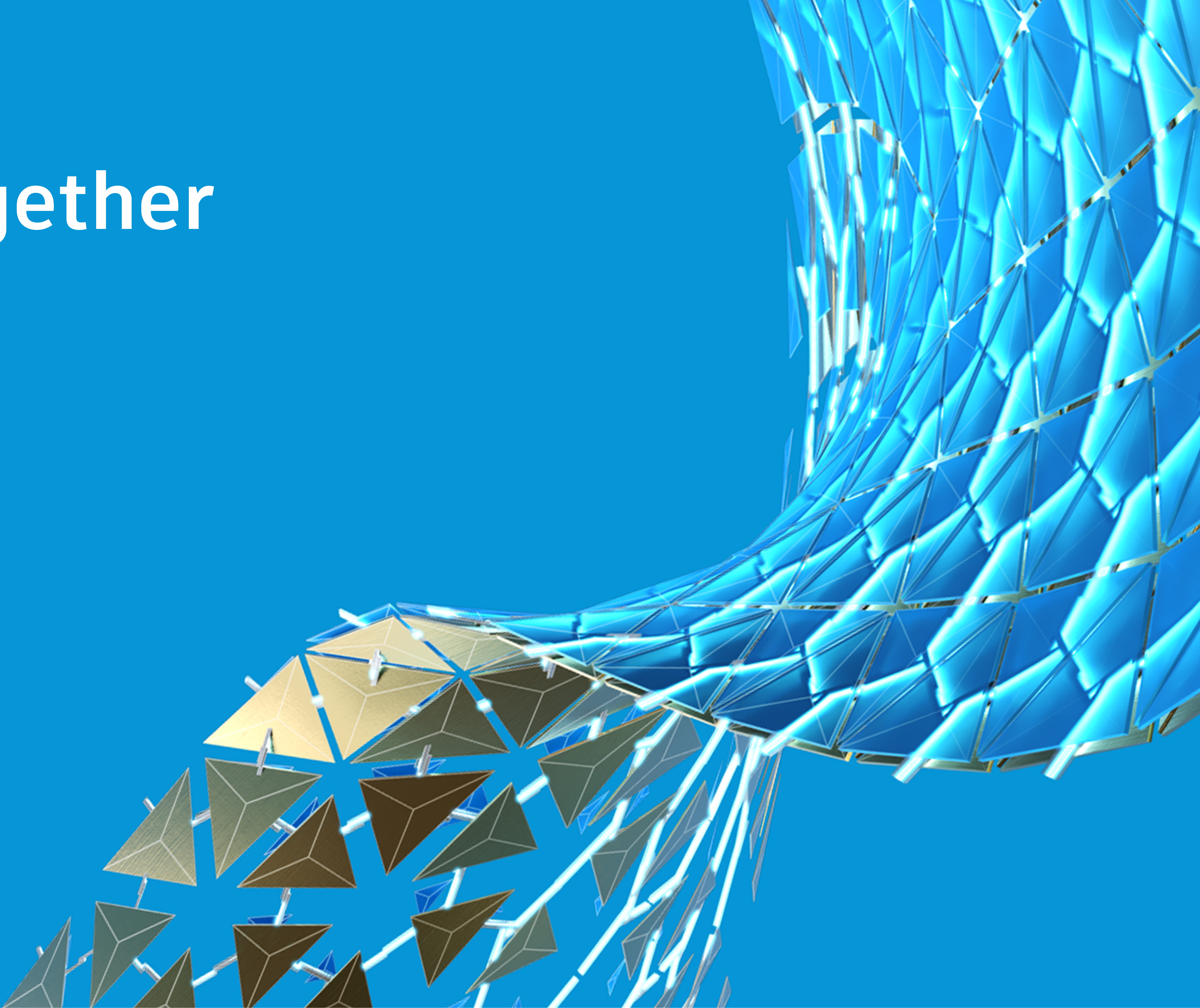
Selected Part: StockMaterial

77.09

```
< ▸ [255]
> let orange                          5);
  NOP_VIEWER.
< undefined
> NOP_VIEWER.getSettingsPanel
< ƒ
```

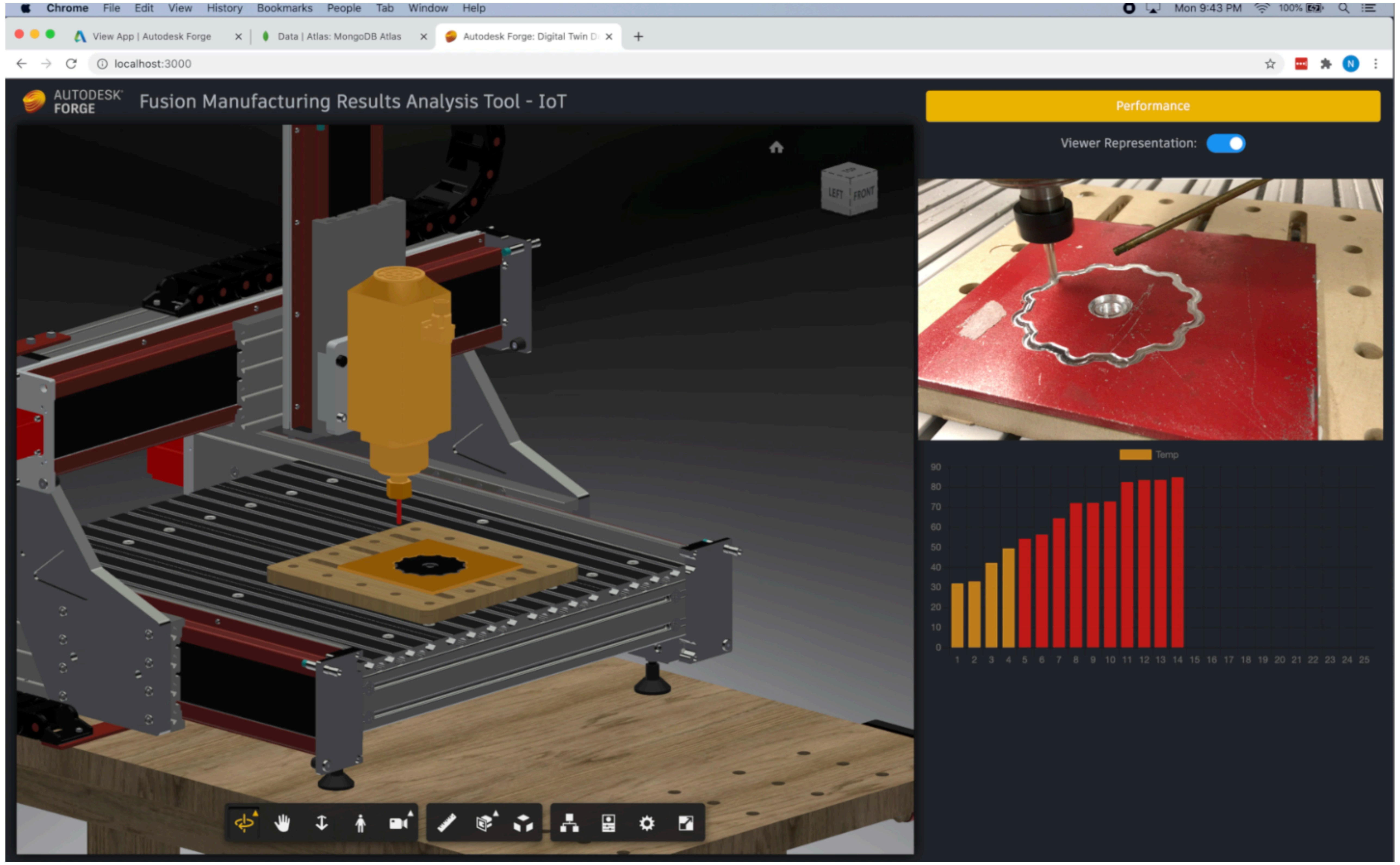
```
getSettingsPanel         GuiViewer3D
getScreenMode            Viewer3D
getScreenModeDelegate
getScreenShot
getSelection
getSelectionCount
getSelectionVisibility
```

Put it all together

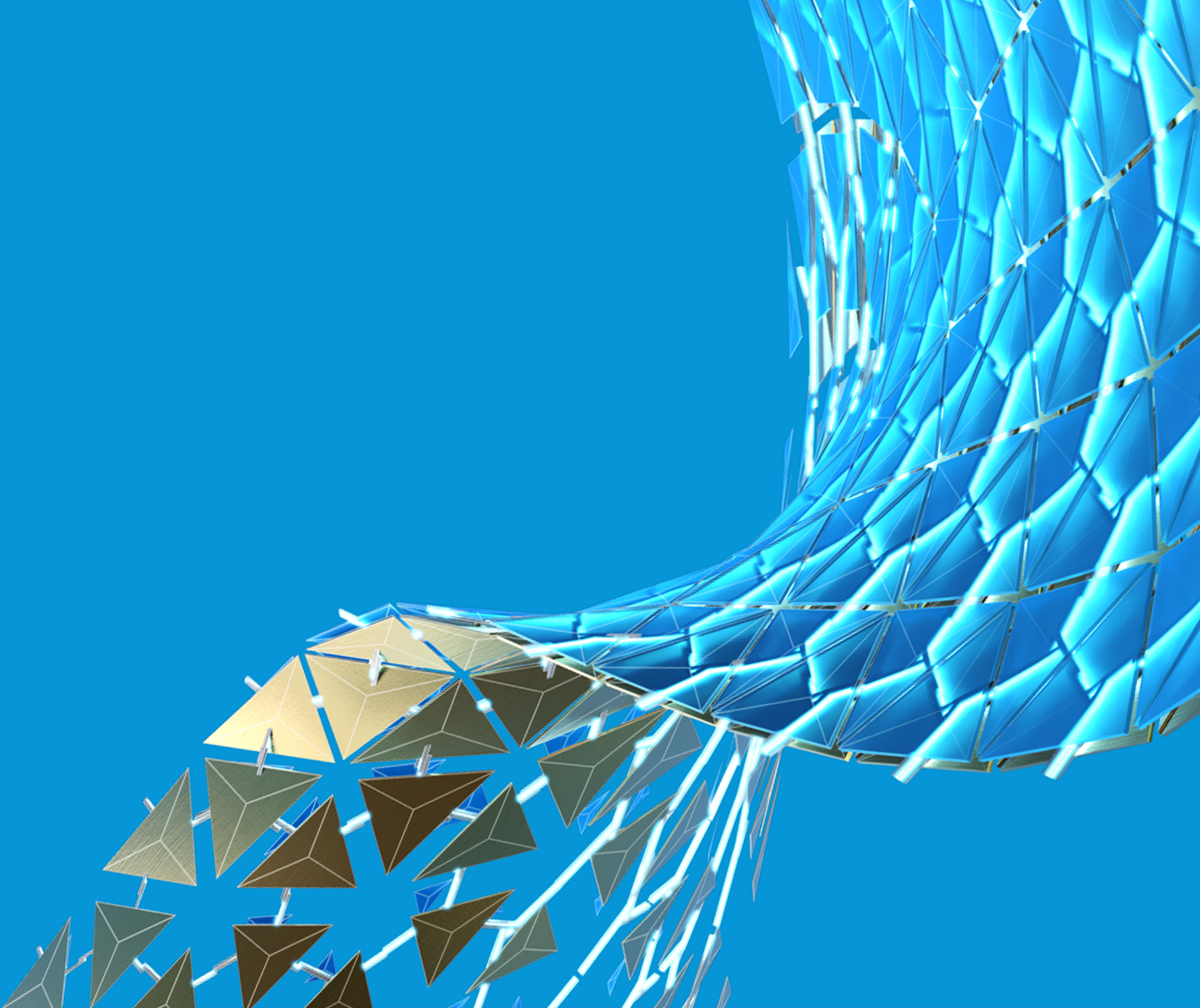


Unit testing MQTT to database

- See class video and handout for details
- Watch the video as we put the demo in action!


Chrome File Edit View History Bookmarks People Tab Window Help
Mon 9:43 PM
100%
View App | Autodesk Forge
Data | Atlas: MongoDB Atlas
Autodesk Forge: Digital Twin D
localhost:3000
AUTODESK FORGE
Fusion Manufacturing Results Analysis Tool - IoT
LEFT
FRONT
Performance
Viewer Representation:
Temp
90
80
70
60
50
40
30
20
10
0
1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25

Wrap-up



We Discussed

- Deploy database
- Build the MQTT broker
- Build the IoT sensor
- Build the IoT firmware
- Test MQTT → Database
- Translate our Fusion 360 model
- Build Forge visualization service
- Put it all Together
- Wrap things up

Our Solution Recipe

Visualization Engine	**Forge Viewer**
Database	**MongoDB**
DB Protocol	**MongoDB+SRV**
IoT Protocol	**MQTT**
IoT Device	**ESP8266 NodeMCU ESP-12E**
Temperature Sensor	**DS18B20**

Digital Resources and References

- **Download this Sample Project >** https://github.com/Autodesk-Forge/forge-iot-cnc
- **Forge API Documentation** > https://forge.autodesk.com/developer/documentation
- **Forge Viewer >** https://forge.autodesk.com/en/docs/viewer/v7/overview/
- **MongoDB Atlas >** https://docs.atlas.mongodb.com/
- **NodeMCU ESP8266 >** https://www.instructables.com/NodeMCU-ESP8266-Details-and-Pinout/
- **MQTT >** https://mqtt.org/mqtt-specification/
- **Digital Thermometer (DS18B20) >** https://datasheets.maximintegrated.com/en/ds/DS18B20.pdf

AUTODESK®