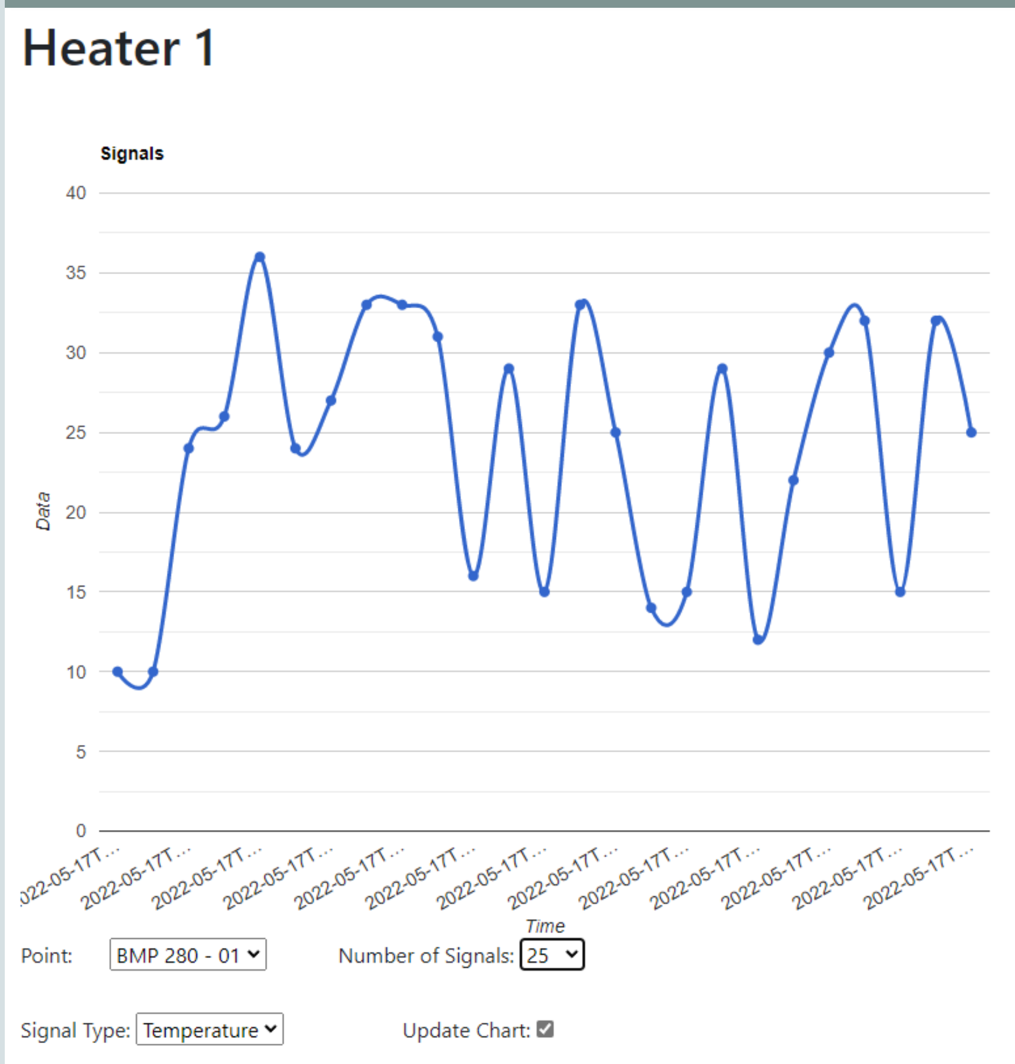
University of South-Eastern Norway

Development of Open Source Datalogging and Monitoring Resources for IoT Platform

Håkon Helgesen
MSc. Industrial IT and Automation

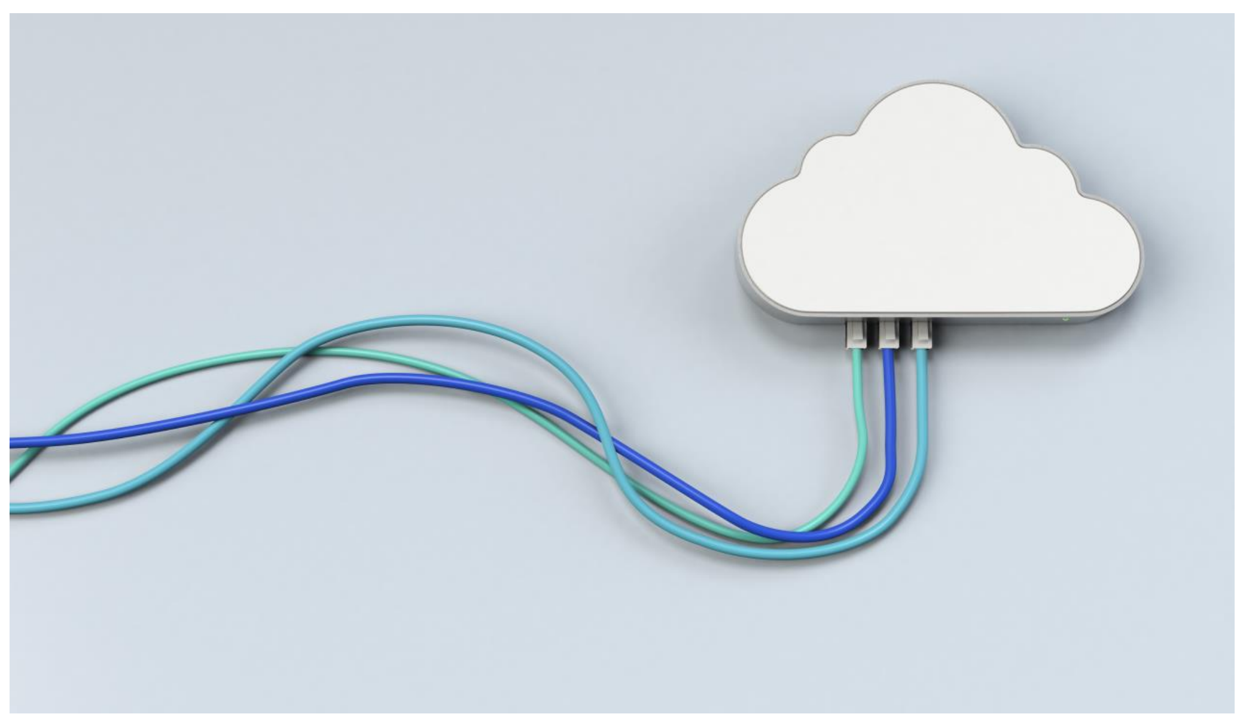


Heater 1

Point: BMP 280 - 01
Number of Signals: 25
Signal Type: Temperature
Update Chart: ☑

Industry 4.0 and IoT

- Internet enabled sensors and controllers
- Data collection
- Data organization
- Prototyping with SBCs



Dimension Four and GraphQL

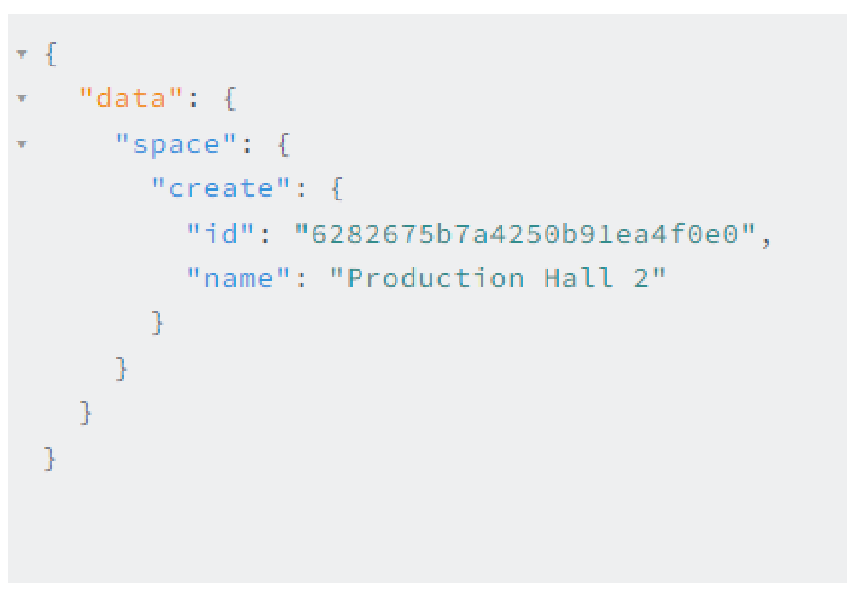
- Custom Data Storage
- GraphQL
    - Recieve only queried information
    - Save bandwidth
- Grenland, Norway

Accessibility

- How do we make a GraphQL service available for beginner to intermediate users, without the need for GraphQL-knowledge?
- What programming languages are most likely to be used?
- How do we visualize collected data and make administrative actions user friendly?

Planning and Development

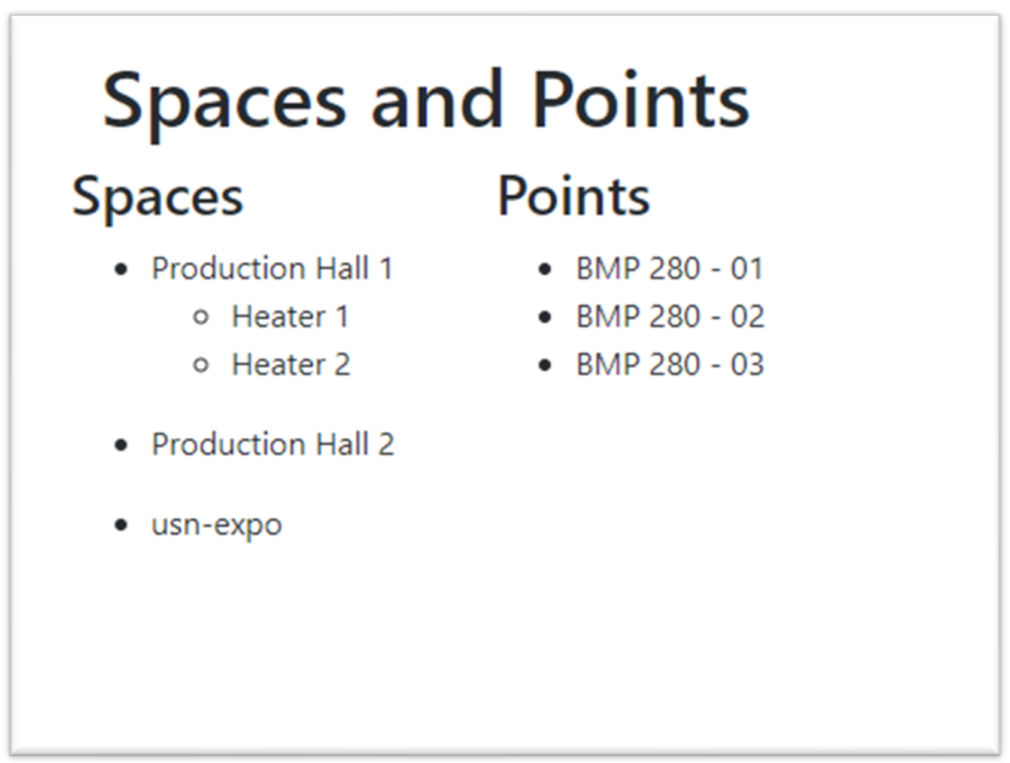
- Communication over HTTP
- Python
- Arduino
- ASP.net Core



```
{
  "data": {
    "space": {
      "create": {
        "id": "6282675b7a4250b91ea4f0e0",
        "name": "Production Hall 2"
      }
    }
  }
}
```

The Product

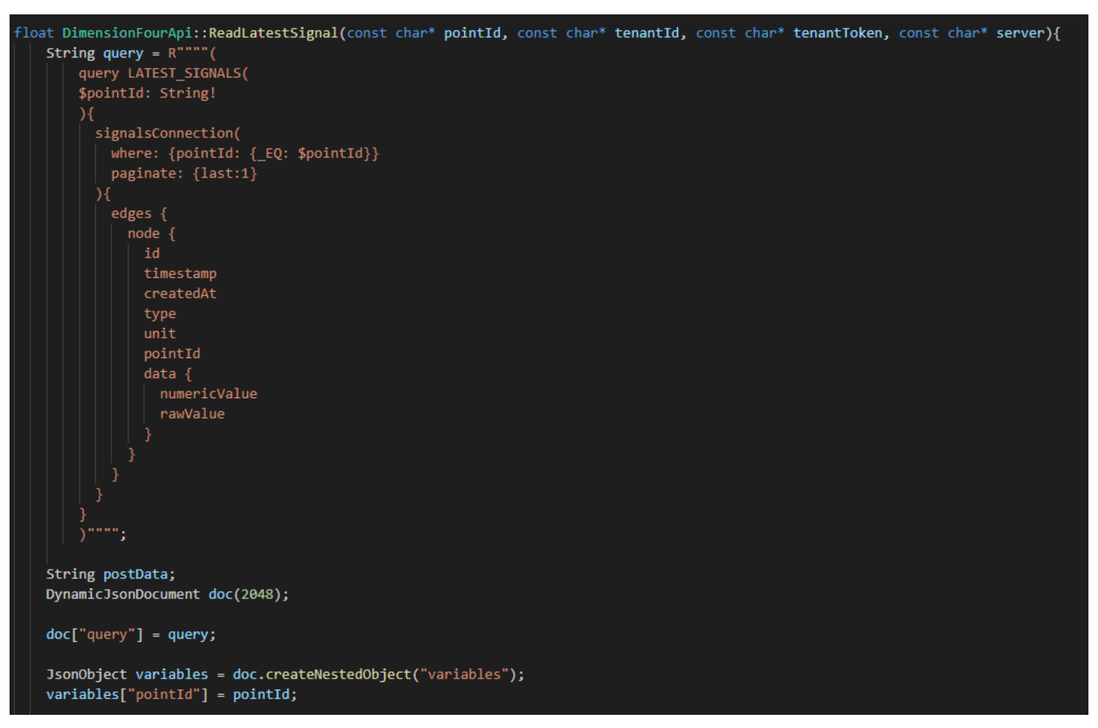
- Python and Arduino Libraries
- ASP.net web-application for administration and monitoring



Spaces and Points

Spaces
- Production Hall 1
    - Heater 1
    - Heater 2
- Production Hall 2
- usn-expo

Points
- BMP 280 - 01
- BMP 280 - 02
- BMP 280 - 03

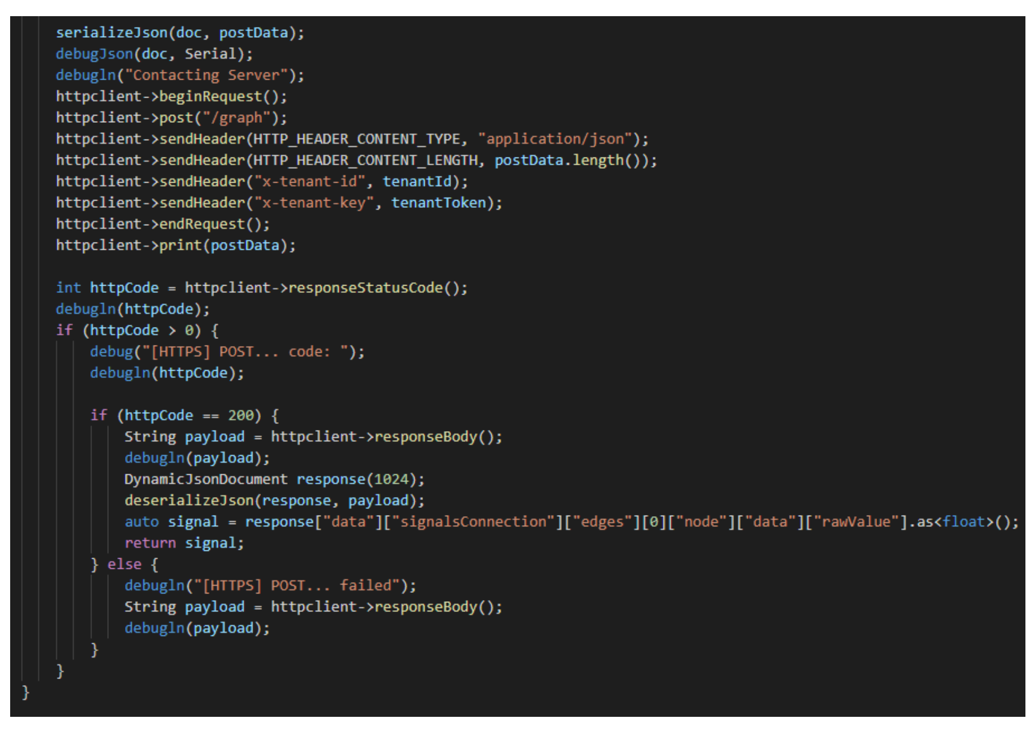


```
float DimensionFourApi::ReadLatestSignal(const char* pointId, const char* tenantId, const char* tenantToken, const char* server){
    String query = R""""(
        query LATEST_SIGNALS(
        $pointId: String!
        ){
          signalsConnection(
            where: {pointId: {_EQ: $pointId}}
            paginate: {last:1}
          ){
            edges {
              node {
                id
                timestamp
                createdAt
                type
                unit
                pointId
                data {
                  numericValue
                  rawValue
                }
              }
            }
          }
        }
        )"""";

    String postData;
    DynamicJsonDocument doc(2048);

    doc["query"] = query;

    JsonObject variables = doc.createNestedObject("variables");
    variables["pointId"] = pointId;
```



```
    serializeJson(doc, postData);
    debugJson(doc, Serial);
    debugln("Contacting Server");
    httpclient->beginRequest();
    httpclient->post("/graph");
    httpclient->sendHeader(HTTP_HEADER_CONTENT_TYPE, "application/json");
    httpclient->sendHeader(HTTP_HEADER_CONTENT_LENGTH, postData.length());
    httpclient->sendHeader("x-tenant-id", tenantId);
    httpclient->sendHeader("x-tenant-key", tenantToken);
    httpclient->endRequest();
    httpclient->print(postData);

    int httpCode = httpclient->responseStatusCode();
    debugln(httpCode);
    if (httpCode > 0) {
        debug("[HTTPS] POST... code: ");
        debugln(httpCode);

        if (httpCode == 200) {
            String payload = httpclient->responseBody();
            debugln(payload);
            DynamicJsonDocument response(1024);
            deserializeJson(response, payload);
            auto signal = response["data"]["signalsConnection"]["edges"][0]["node"]["data"]["rawValue"].as<float>();
            return signal;
        } else {
            debugln("[HTTPS] POST... failed");
            String payload = httpclient->responseBody();
            debugln(payload);
        }
    }
}
```

Supervisors:
Hans-Petter Halvorsen, Daniel Warholm

In collaboration with
Dimension Four AS