

Softwareentwurf und Anwendungen verteilter Systeme

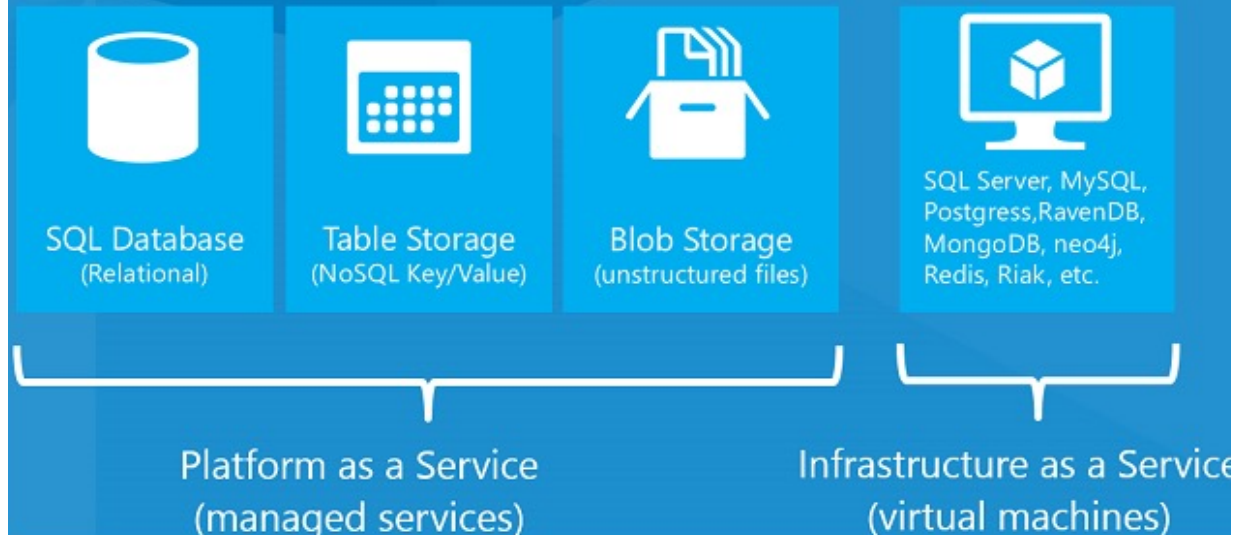
Digital Product Design and Development B.A.

Semester 3

Hochschule für Gestaltung Schwäbisch Gmünd

Dozent: Yannick Schiele

Data Storage Options on Windows Azure



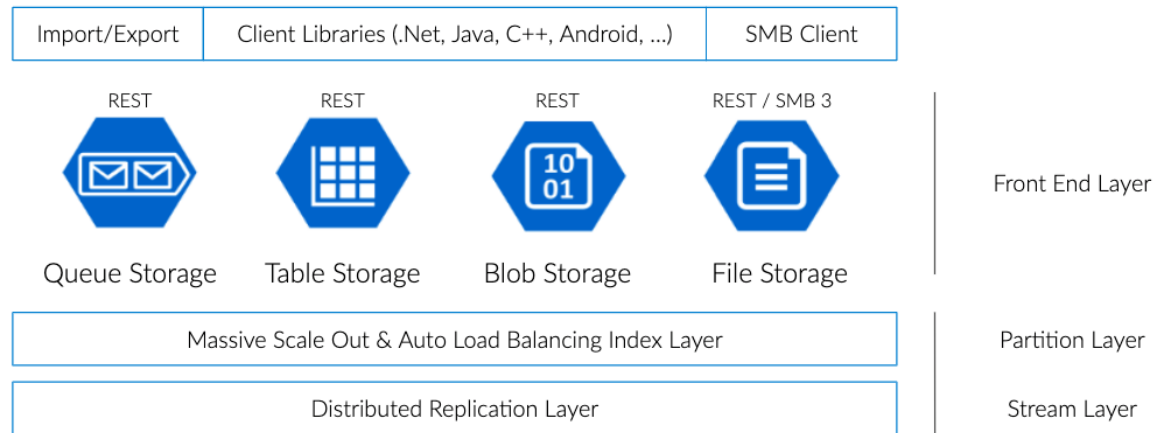
<https://learn.microsoft.com/de-de/training/modules/get-started-cloud-storage-for-iot/2-introduction-lambda-architecture>

Azure IoT Cloudspeicheroptionen

Folgende Azure Storage-Optionen werden häufig in IoT-Lösungen verwendet:

- Azure Blob Storage, als Azure IoT Hub-Routingendpunkt verfügbar
- Azure SQL-Datenbank, als Azure Stream Analytics-Output verfügbar

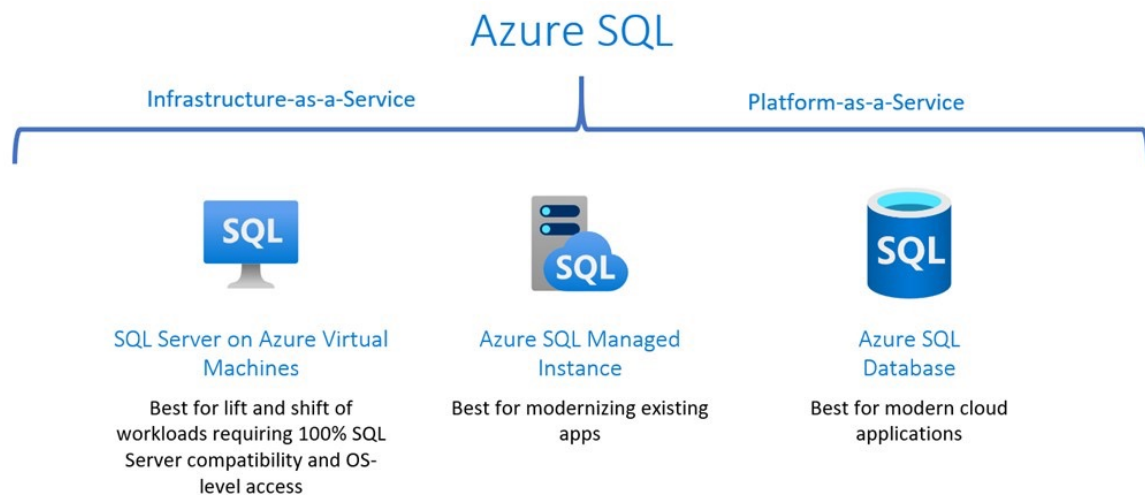
Azure Storage Architecture



<https://learn.microsoft.com/de-de/training/modules/get-started-cloud-storage-for-iot/2-introduction-lambda-architecture>

Speicherkontentypen

- File Storage
 - ermöglicht die Einrichtung hochverfügbarer Netzwerkdateifreigaben, auf die über das standardmäßige SMB-Protokoll (Server Message Block) zugegriffen werden kann
- Blob Storage
 - Objektspeicherlösung für die Speicherung großer Mengen von unstrukturierten Daten, z.B. Text oder Binärdaten, optimiert
- Queue Storage
 - wird zum Speichern und Abrufen von Nachrichten verwendet. Nachrichten können eine Größe von bis zu 64 KB haben, und eine Warteschlange kann Millionen von Nachrichten enthalten



<https://learn.microsoft.com/de-de/training/modules/get-started-cloud-storage-for-iot/2-introduction-lambda-architecture>

Azure SQL-Datenbank

- Azure SQL-Datenbank ist eine vollständig verwaltete PaaS-Datenbank-Engine (Platform-as-a-Service), bei der die meisten Funktionen für die Datenbankverwaltung, z.B. Upgrades, Patches, Sicherungen und Überwachung, ohne Benutzereingriff erfolgen
- Azure SQL-Datenbank wird in der aktuellen stabilen Version von SQL Server-Datenbank-Engine und gepatchtem Betriebssystem mit 99,99% Verfügbarkeit ausgeführt

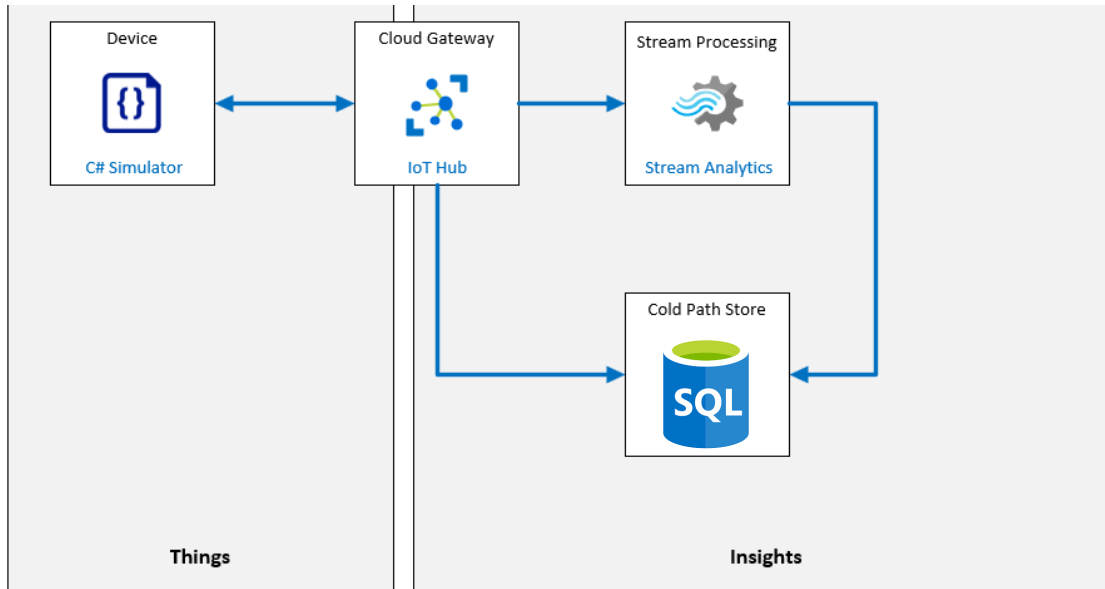


<https://learn.microsoft.com/de-de/training/modules/get-started-cloud-storage-for-iot/2-introduction-lambda-architecture>

Stream Analytics

Azure Stream Analytics bietet eine cloudbasierte Streamverarbeitungs-Engine, mit der ein Echtzeitdatenstrom aus verschiedenen Quellen gefiltert, aggregiert und verarbeitet werden kann.

Die Ergebnisse dieser Verarbeitung lassen sich dann verwenden, um automatisierte Aktivitäten durch eine Anwendung auszulösen oder Echtzeitvisualisierungen zu generieren



Datenbank IoT Hub Beispiel

Gemeinsame Programmierung:

- SQL Server
- SQL Datenbank
 - IP Adresse hinzufügen
- Stream Analytics Job
 - Input
 - Output
 - Query

Azure Ressourcen

Für Webentwicklung

Azure App Service



Web Apps



Mobile Apps



API Apps



Logic Apps

<https://learn.microsoft.com/en-us/training/modules/host-a-web-app-with-azure-app-service/>

Azure App Service

Azure App Service ist eine vollständig verwaltete Hosting-Plattform für Webanwendungen.

Dieser von Azure angebotene Plattform als Service (PaaS) ermöglicht es den Nutzern, sich auf das Design und die Erstellung der App zu konzentrieren, während Azure sich um die Infrastruktur zur Ausführung und Skalierung der Anwendungen kümmert.

Home > App Services >

Create Web App ...

Basics Deployment Networking Monitoring Tags Review + create

App Service Web Apps lets you quickly build, deploy, and scale enterprise-grade web, mobile, and API apps running on any platform. Meet rigorous performance, scalability, security and compliance requirements while using a fully managed platform to perform infrastructure maintenance. [Learn more](#)

Project Details

Select a subscription to manage deployed resources and costs. Use resource groups like folders to organize and manage all your resources.

Subscription * ⓘ Azure for Students-YS

Resource Group * ⓘ (New) Resource group

[Create new](#)

Instance Details

Need a database? [Try the new Web + Database experience.](#)

Name * Web App name. .azurewebsites.net

Publish * ☒ Code ☐ Docker Container ☐ Static Web App

Runtime stack * Select a runtime stack

Operating System ☒ Linux ☐ Windows

Region * Central US

i Not finding your App Service Plan? Try a different region or select your App Service Environment.

Review + create < Previous Next : Deployment >

Erstellung Web App im Azure Portal

Deployment der Web App mit VSCode

Node js App:

npm init

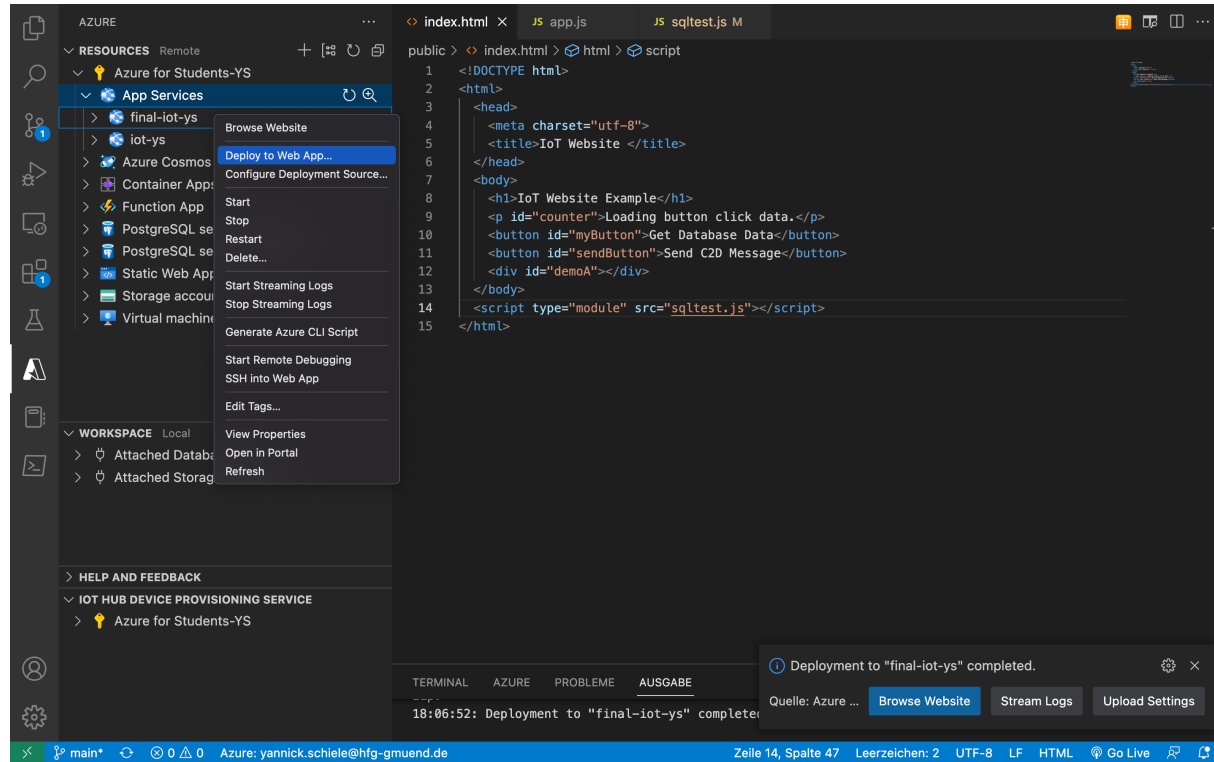
npm install azure-iot-hub –save

npm install mssql

npm install express –save

Lokal ausführen:

node app.js



Datenbank Abfrage

Mit einer Web-Applikation

```
const config = {
  user: 'yannick',
  password: '',
  server: 'test-db-server-2.database.windows.net',
  port: 1433, // optional, defaults to 1433,
  database: 'test-vorlesung-db',
  authentication: { type: 'default' },
  options: { encrypt: true }
}
```

```
app.get('/', function (req, res) {
  res.sendFile(__dirname + '/index.html'); });
app.get('/db', (req, res) => { // connect to the database
  sql.connect(config, function (err) {
    if (err) console.log(err);
    // create Request object
    var request = new sql.Request();
    // query to the database and get the records
    request.query('SELECT TOP (1000) * FROM
    [dbo].[testDevice]', function (err, recordset) {
      if (err) console.log(err)
      res.status(200).json(recordset);
    });
  });
});
```

<https://learn.microsoft.com/de-de/azure/azure-sql/database/connect-query-nodejs?view=azuresql&tabs=macos>

Code der Abfrage

Beispiel Code für die Programmierung der Server API und des Client Aufruf

Cloud-to-Device Messages

Mit einer Web-Applikation

```

app.post('/sendMessage', (req, res) => {
  var Client = require('azure-iot-hub').Client;
  var Message = require('azure-iot-common').Message;
  var connectionString = ""
  var targetDevice = "storageDevice";
  var client = Client.fromConnectionString(connectionString);

  client.open(function (err) {
    if (err) { console.error('Could not connect: ' + err.message); }
    else { console.log('Client connected'); }

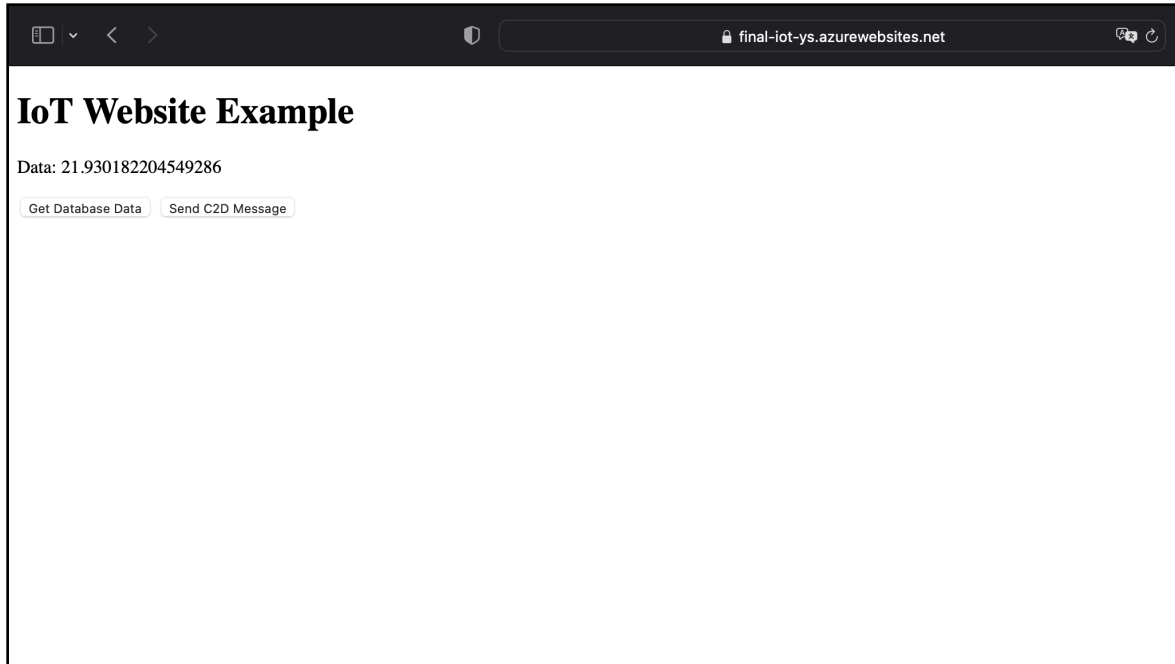
    // Create a message and send it to the IoT Hub
    var data = JSON.stringify({ text: 'food123456' });
    var message = new Message(data);
    console.log('Sending message: ' + message.getData()); client.send(targetDevice, message,
    printResultFor('send')); } });

    // Helper function to print results in the console
    function printResultFor(op) {
    return function printResult(err, resLokal) {
      if (err) { console.log(op + ' error: ' + err.toString()); }
      else {
        console.log(op + ' status: ' + resLokal.constructor.name);
        res.status(201).json(op + ' status:' + resLokal.constructor.name);
      }
    };
  };
}

```

Senden einer C2D-Nachrichten

Beispiel Code für die Programmierung der Server API und des Client Aufruf



Ausgangslage

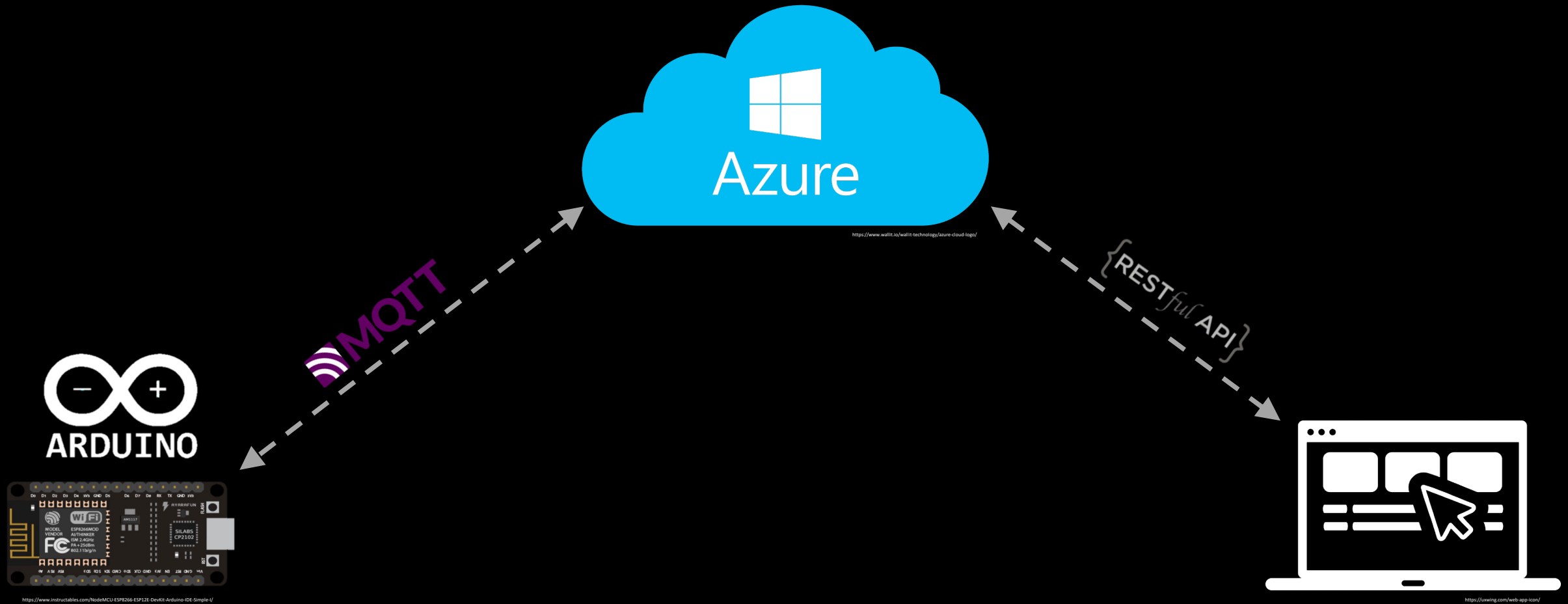
Node JS Webserver in Azure

Kommunikation über REST APIs

Javascript Client WebApp mit

- Zugriff auf Datenbank
- C2D Messages

Finales Projekt



Bewertung

- Projekte (pass/fail)
 - Kein auskommentierter Code
 - Der Code ist lauffähig auf eurem spezifizierten Mobile Device, im neusten Chrome/Firefox und auf dem ESP32/ESP8266
 - Software Struktur und Architektur - Dokumentation der APIs
- Projektbericht
- Abgabe über Github Classroom
- Designaufgabe
 - Keine losen Kabel
 - Gehäuse
 - UI
- Semesterausstellung: Review in der Woche davor

Gruppenarbeit