

IoT-Labor: Smart Lock

Dokumentation

Bachelor of Science

des Studiengangs Informatik

an der Dualen Hochschule Baden-Württemberg Stuttgart

von

Tom Freudenmann, Maximilian Nagel, Marcel Fleck

26.04.2023

Bearbeitungszeitraum
Matrikelnummern, Kurs
Dozent

10.03. - 26.04.2023
6378195, 7362334, 9611872, INF20D
Hartmut Seitter

Selbstständigkeitserklärung

Ich versichere hiermit, dass ich meine Dokumentation mit dem Thema: *IoT-Labor: Smart Lock* selbstständig verfasst und keine anderen als die angegebenen Quellen und Hilfsmittel benutzt habe. Ich versichere zudem, dass die eingereichte elektronische Fassung mit der gedruckten Fassung übereinstimmt.

Stuttgart, 26.04.2023

Tom Freudenmann, Maximilian Nagel, Marcel Fleck

Inhaltsverzeichnis

Abkürzungsverzeichnis	III
Abbildungsverzeichnis	IV
1 Einleitung	1
2 Architektur	2
2.1 Device-Layer	2
2.2 Network-Layer	2
2.3 Service-Layer	3
2.4 Application-Layer	4
3 Ausblick	5

Abkürzungsverzeichnis

HTTP	Hypertext Transfer Protokol
IoT	Internet of Things
JSON	JavaScript Objective Notation
MQTT	Message Queuing Telemetry Transport
TTN	The Things Network

Abbildungsverzeichnis

2.1	Architektur-Diagramm des Smart-Locks	2
2.2	Node-Red Serverarchitektur für das Smart-Lock	3

1 Einleitung

TODO: Hier anfangen zu schreiben: Buisnesscase

2 Architektur

TODO: Hier anfangen zu schreiben

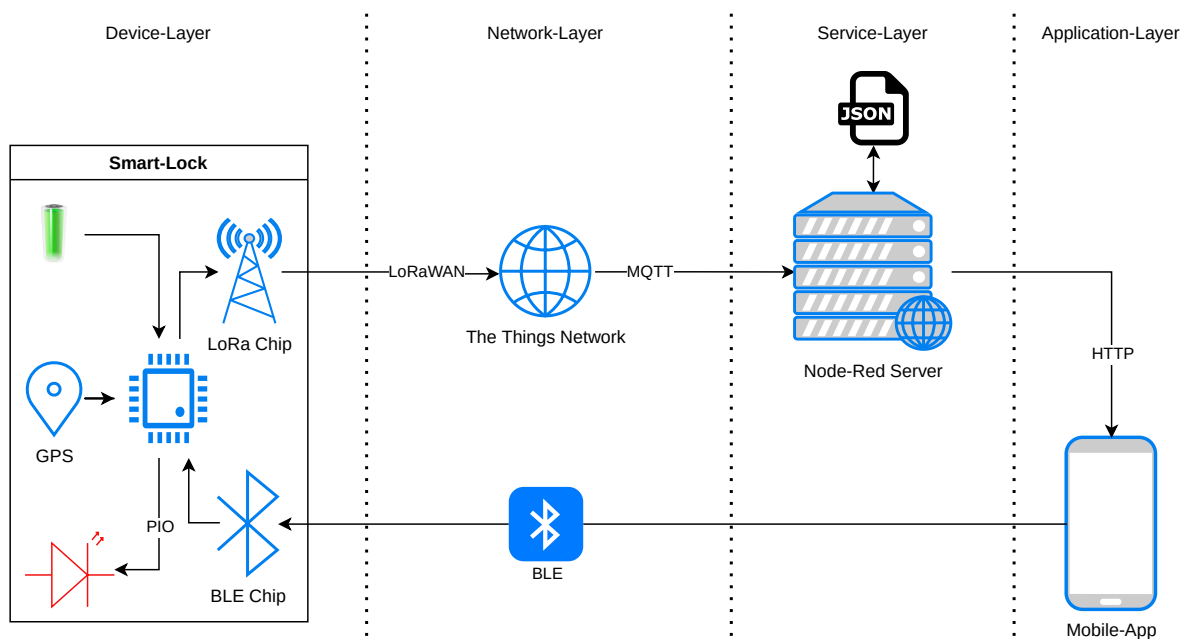


Abbildung 2.1: Architektur-Diagramm des Smart-Locks

2.1 Device-Layer

TODO: Hier anfangen zu schreiben

2.2 Network-Layer

TODO: Hier anfangen zu schreiben

2.3 Service-Layer

Das Service-Layer besteht aus einem Node-Red Server, der in einem Docker-Container läuft. Das hat den Vorteil, das die Anwendung beliebig umgezogen oder skaliert werden kann. Zusätzlich empfängt der Server die Daten aus dem The Things Network (TTN) über Message Queuing Telemetry Transport (MQTT) und speichert sie local in der *Context*-Variablen und persistent als JavaScript Objective Notation (JSON)-Dokument ab. Dadurch können die Daten auch nach einem Neustart der Anwendung weiter verwendet werden. Abbildung 2.2 zeigt die folgenden drei Prozesse in Node-Red:

- **Initialisierung:** Der erste *Flow*, der beim Serverstart einmalig ausgeführt wird. Er lädt das gespeicherte JSON-Dokument und setzt die *Context*-Variablen auf die gespeicherten Werte.
- **Verbindung zum TTN-Server:** Dieser Prozess wird für jede im TTN empfangene und weitergeleitete Nachricht ausgeführt und speichert die empfangenen **GPS!** (**GPS!**)- und Sensor-Informationen im JSON-Dokument bzw. Context des Servers.
- **App Anfrage:** Für jede empfangene Hypertext Transfer Protokol (HTTP)-Anfrage an die Route *server-adresse:1880/device-id* wird der *Flow* ausgeführt. Er lädt die Daten für die mitgelieferte *device-id* und schickt sie als Antwort zurück and die App.

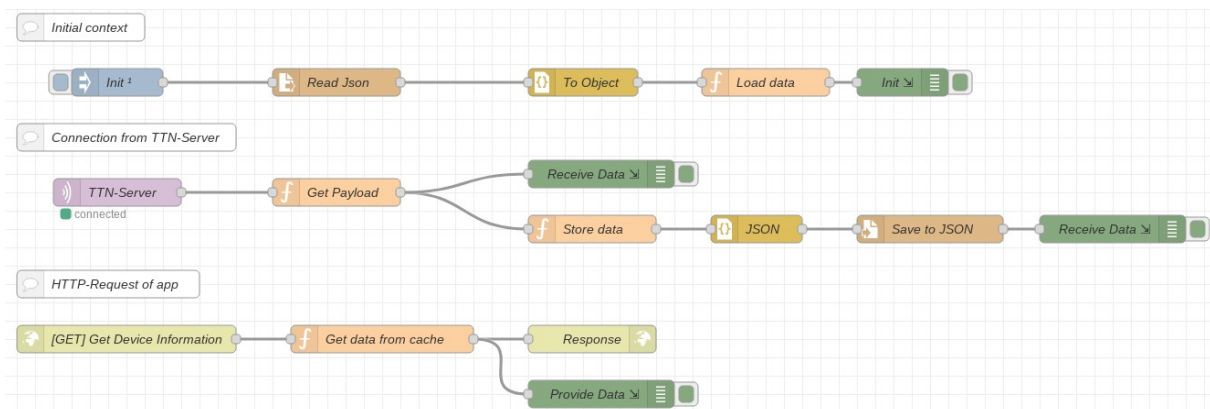


Abbildung 2.2: Node-Red Serverarchitektur für das Smart-Lock

Außerdem wird der Server in der Azure-Cloud, auf einem Ubuntu-Server gehostet. Dadurch ist der Server aus dem Internet zugänglich und die Daten können jeder Zeit vom Benutzer abgerufen werden.

Zusammengefasst, deckt der Server das Service-Layer mit persistenter Speicherung ab, indem er Nachrichten aus dem Network-Layer empfängt und an das Application-Layer weiterleitet. Zusätzlich werden die Daten im Server gespeichert, um sie jeder Zeit abfragen

zu können, auch wenn der Server neu gestartet wurde. In Zukunft lässt sich das System weiter skalieren oder auf einem eigenen Server hosten, da es in einem Docker-Container läuft.

2.4 Application-Layer

TODO: Hier anfangen zu schreiben

3 Ausblick

TODO: Hier anfangen zu schreiben