



*firebrok*

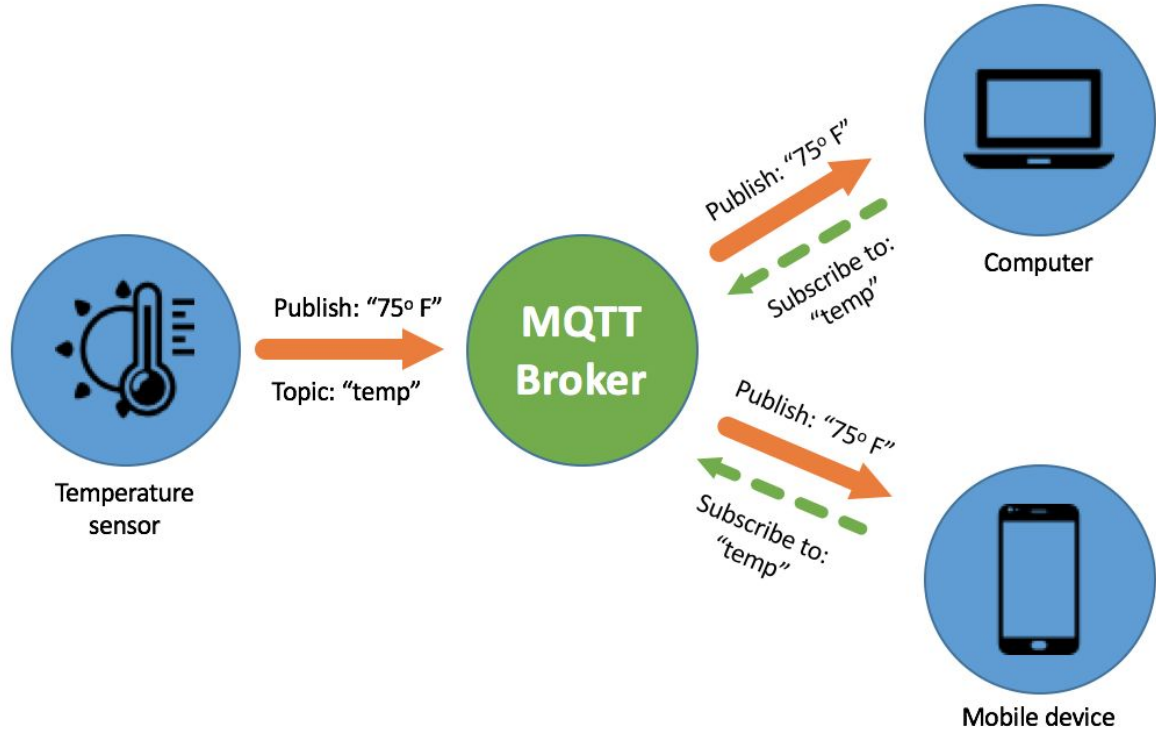
# 1

PROJET



# 2

## PROTOCOLE MQTT



# 3

TECHNOLOGIES  
CHOISIES

## Technologies serveur



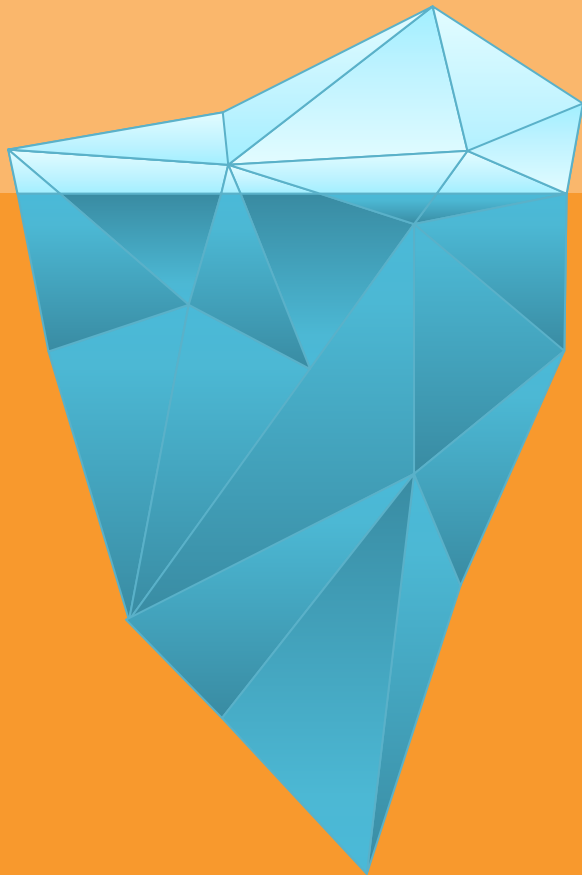
Firebase

## Technologies client



# 4

## INFRASTRUCTURE



### **Dashboards**

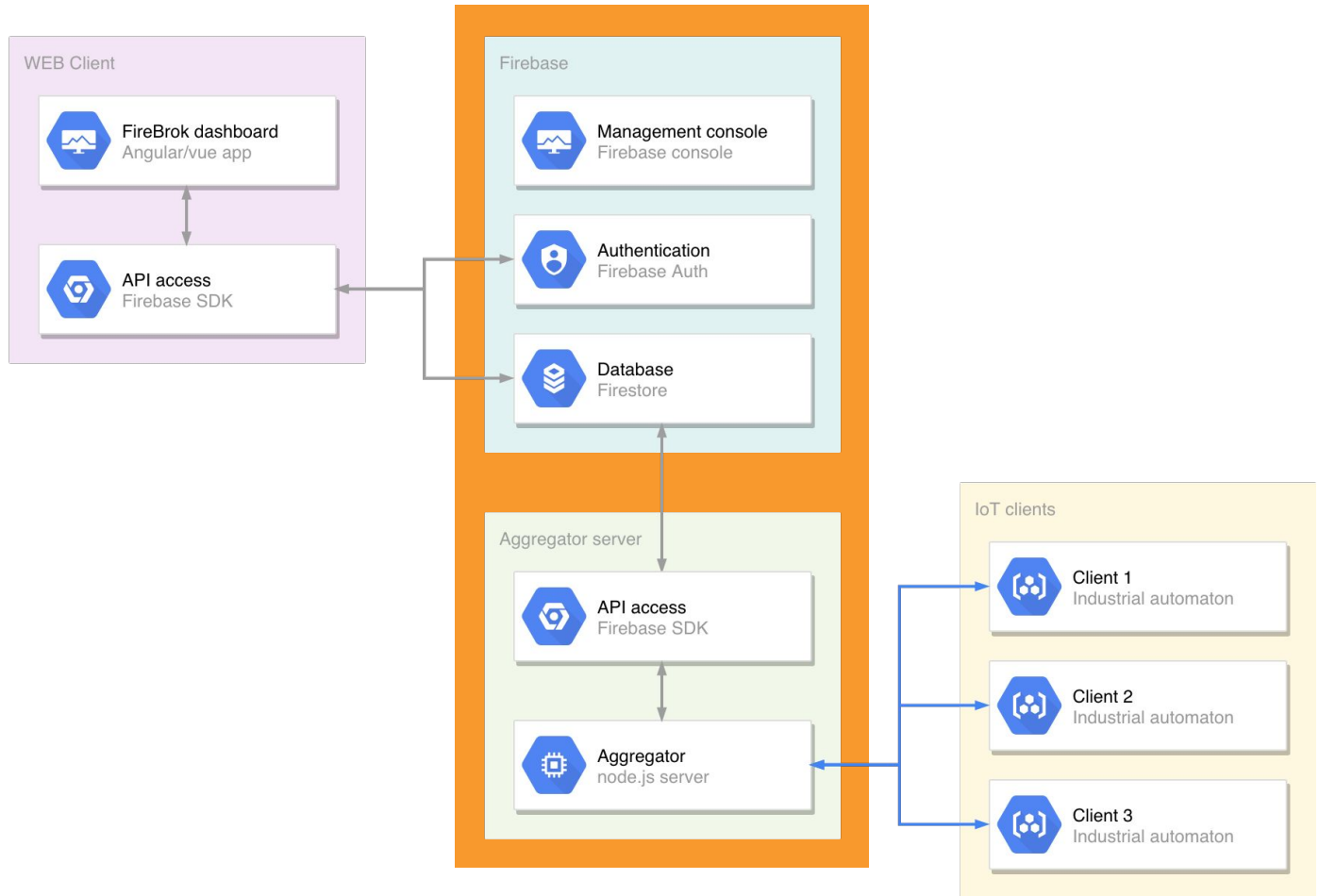
Développé indépendamment de la plateforme.

**Base de données,**  
Firebase

**Aggregateur**  
Serveur node.js

# 4

## INFRASTRUCTURE



# 5

## AGGREGATEUR FONCTIONNEMENT MQTT



```
var net = require('net')
var mqttCon = require('mqtt-connection')
var server = new net.Server()

server.on('connection', function (stream) {
  var client = mqttCon(stream)

  // Automaton connected
  client.on('connect', function (packet) {

    var automatonName = packet.username
    var automatonId = packet.password.toString('utf8')

  })
})

// Listen on port 1883
server.listen(1883)

console.log("Listening ... ")
```

# 5

## AGGREGATEUR FONCTIONNEMENT FIREBASE

```
var admin = require("firebase-admin");

var serviceAccount = require("../token/firebrok-adminsdk.json");

admin.initializeApp({
  credential: admin.credential.cert(serviceAccount),
  databaseURL: "https://firebrok.firebaseio.com"
});

var db = admin.firestore();
```



# 5

## AGGREGATEUR FONCTIONNEMENT AUTORISATIONS

```
automatonsCol.doc(automatonId).get()
  .then(doc => {
    if (!doc.exists || doc.data().name !== automatonName) {
      console.log(`Automaton ${automatonName} refused`)
      client.destroy()
    } else {
      console.log(`Automaton ${automatonName} authorized`)

      automatonsCol.doc(automatonId).set({
        connected: true
      }, { merge: true })
      authorized = true
    }
  })
  .catch(err => {
    console.log(err)
    client.destroy()
  })
```

# 5

## AGGREGATEUR FONCTIONNEMENT MESSAGES

```
let fluxDoc = automationsCol
    .doc(automatonId)
    .collection("topics")
    .doc(topic)
    .collection("flux")
    .doc()

fluxDoc.set({
  message: packet.payload.toString('utf8'),
  timestamp: admin.firestore.Timestamp.fromDate(currentDate)
}, { merge: false });

console.log(`Automaton ${automatonName} send :`)
console.log(`Topic : ${topic}`)
console.log(`Message : ${packet.payload.toString('utf8')}`)
```

# 5

AGGREGATEUR  
TEST  
SCRIPT PYTHON



```
from paho.mqtt.client import Client

# Create the mqtt client
mqtt_client = Client(username)

# Assign events
mqtt_client.on_connect = on_connect
mqtt_client.on_disconnect = on_disconnect

# Connexion
mqtt_client.connect('endpoint')

# Publish messages
mqtt_client.publish('mon topic', 'mes données')
```

# 5

AGGREGATEUR  
TEST  
MQTT LENS



## Add a new Connection ×

### Connection Details

Connection name

local

Connection color scheme



Hostname

tcp://

localhost

Port

1883

Client ID

lens\_7IVvzscyav9X1ObnlrDVAtDhQGE

Generate a random ID

Session

☒ Clean Session

Automatic Connection

☒ Automatic Connection

Keep Alive

120

seconds

### Credentials

Username

MQTT Lens

Password

\*\*\*\*\*

# 5

AGGREGATEUR  
TEST  
MQTT LENS



MQTTlens

Version 0.0.14

Connections + ^

local



Connection: local

Subscribe



Publish



topic

0 - at most once



Retained

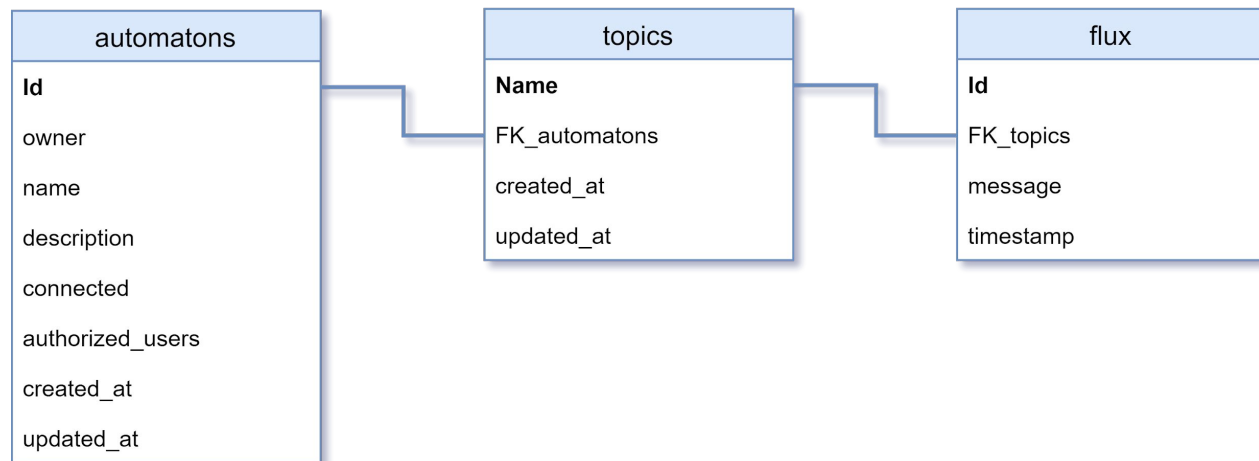
PUBLISH

Message



# 6

## FIREBASE BASE DE DONNÉES



# 6


## FIREBASE BASE DE DONNÉES




```
{
  "automatons": {
    "HD3JKT7Cmld93BB1QVHV": {
      "owner": "msDGeAqwfOPSDSwFUxxQgdWm5352",
      "name": "Mon automate",
      "description": "Il est dans la salle C333",
      "connected": "true",
      "authorized_users": [
        "khLsPy9tjxRJt4GLW0uij1rKMNu2",
        "9EljJmOY6Yg5CqiyyZPUVUEUR382"
      ],
      "created_at": "20 mars 2019 à 09:04:38 UTC+1",
      "updated_at": "21 mars 2019 à 10:25:08 UTC+1",
      "topics": {
        "ultrasonic": {
          "name": "Ultrasonic",
          "created_at": "20 mars 2019 à 09:05:55 UTC+1",
          "updated_at": "21 mars 2019 à 10:25:08 UTC+1",
          "flux": {
            "08WfqmCR82hgfbS3DDuq": {
              "message": "51.68",
              "timestamp": "20 mars 2019 à 09:05:55 UTC+1"
            },
            "0A41ZrMipalJnEfy18q0": {
              "message": "60.38",
              "timestamp": "21 mars 2019 à 10:25:08 UTC+1"
            }
          }
        }
      }
    }
  }
}
```







# 6

## FIREBASE AUTHENTIFICATION


 **Firebase**

[Project Overview](#) 

**Développer**


-  **Authentication**
-  Database
-  Storage
-  Hosting
-  Functions
-  ML Kit



**Qualité**


-  Crashlytics

**Spark**  
Gratuit  
0 \$/mois


**Changer de formule**



FireBrok 






[Accéder à la documentation](#)  




**Authentication** [Configuration Web](#) 

[Utilisateurs](#) [Mode de connexion](#) [Modèles](#) [Utilisation](#)

 Rechercher par adresse e-mail, numéro de téléphone ou ID utilisateur

[Ajouter un utilisateur](#)  

Identifiant	Fournisseurs	Date de création	Dernière connexion	ID utilisateur 
admin@test.ch		12 févr. 2019		4L8BHK1IrtSiliKifsNbSuisJDh2
kevin.jordil@cpnv.ch		5 mars 2019	20 mars 2019	9ElJmOY6Yg5CqjyyZPUVUEU...
toto@test.com		21 févr. 2019	20 mars 2019	khLsPy9tjxRJt4GIW0uij1rKMN...
test.test@test.com		1 mars 2019	1 mars 2019	msDGeAqwFOPSDSwFUxxQgd...

Lignes par page : 50  1 - 4 sur 4  



# 6

## FIREBASE AUTHENTICATION

```
import { firebase } from "firebase";

await firebase
  .auth()
  .signInWithEmailAndPassword(this.email, this.password);

firebase
  .auth()
  .onAuthStateChanged(user => {
    this.authenticated = user
  })
```

# 7

## DASHBOARD EXAMPLE VUE



```
<template>
  <ul>
    <li v-for="automaton of automations" :key="automaton.id">
      <p>{{ automaton.name }}</p>
    </li>
  </ul>
</template>

<script>
import { firebase } from 'firebase'

export default {
  firestore() {
    return {
      automations: firebase
        .firestore()
        .collection('automations')
        .orderBy('name', 'desc')
    }
  }
}
</script>
```

# 7

## DASHBOARD EXAMPLE ANGULAR



```
<ul>
  <li *ngFor="let automaton of automatons$ | async">
    <p>{{ automaton.name }}</p>
  </li>
</ul>
```



```
@Component({})
export class MyAwesomeComponent implements OnInit {

  automatons$: Observable<Automaton>;

  constructor(private firestore: AngularFireStore) { }

  ngOnInit() {
    this.automatons$ = this.firestore
      .collection(
        'automatons',
        ref => ref.orderBy('name', 'desc')
      )
      .valueChanges()
  }
}
```



# 8

SYNTHÈSE

Données centralisées

Services découplés

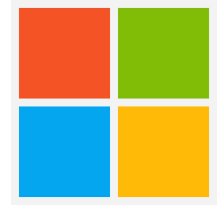
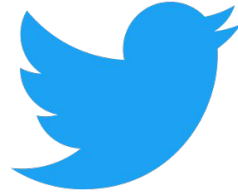
Temps réel

Intégration simple



# 9

**OUVERTURE**  
AUTHENTIFICATION



# 9

## OUVERTURE SYSTÈME DE PERMISSIONS



# 9

OUVERTURE  
FORMAT DES  
DONNÉES

# {JSON}

JavaScript Object Notation



?

# 9

**OUVERTURE**  
BROKER MQTT  
COMPLET







# Questions ?