Software Architecture Design SAD :)

AT70.18

Asian Institute of Technology
School of Engineering and Technology
Assignment 3

Microservice + Serverless Architecture

Submitted to:
Dr. Chaklam Silpasuwanchai

Submitted by:
Abhishek Koirala (st120199)
Bibhuti regmi (st120320)
Shikshya Dahal (st120186)

Submitted Date: 28th March 2019

Role of each member:

We first went through the tutorials in the internet about microservice and how we could implement them. After going through the tutorials, we discussed how we could implement it and what application to make. Then we tried making a microservice in **TypeScript**, using **Firebase**. All of us were equally involved in creating and implementing microservices as well as building JAVA application

Describe the 3rd party microservice you use and why you use it.

We used Firestore as 3rd party microservices from Firebase. Cloud Firestore is a flexible, scalable database for mobile, web, and server development from Firebase and Google Cloud Platform. Like Firebase Realtime Database, it keeps data in sync across client apps through real-time listeners and offers offline support for mobile and web so one can build responsive apps that work regardless of network latency or Internet connectivity. Cloud Firestore also offers seamless integration with other Firebase and Google Cloud Platform products, including Cloud Functions.

Describe your own microservice and why you made it

Our microservice is a medium which connects application to firestore through API without using firestore core function. Normally, to use firestore we need to integrate complete firebase platform in our project which is cumbersome.Our service provides an api in order to insert and read data to and from firestore without integrating firebase.This eases us to use firestore through application in any platform with API backend, without complete integration of firebase to use firestore.

We deployed a typescript project which creates an API endpoint for data entry and retrieval, from FireStore(another microservices by Firebase).

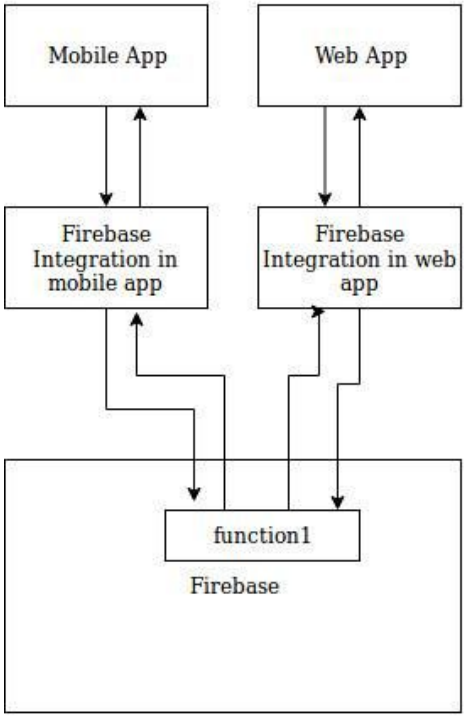

Fig: without our function

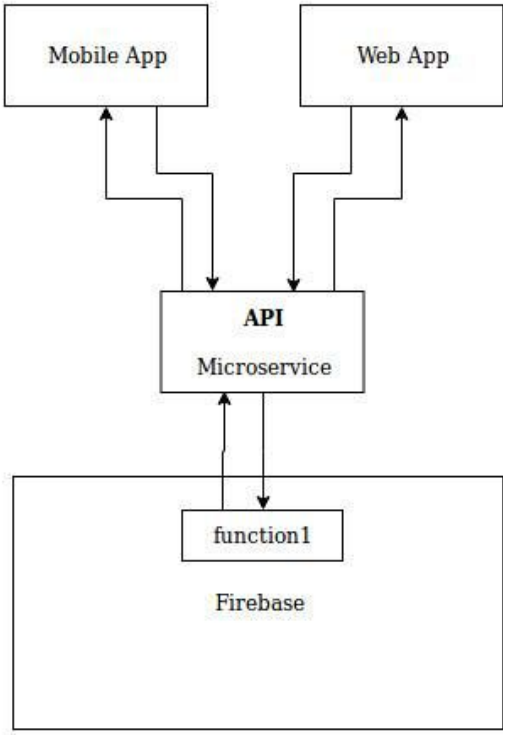

Fig: with our function

Describe your usage of serverless technology

Firebase is a mobile and web application development platform developed by Firebase, Inc. in 2011, then acquired by Google in 2014. As of October 2018, Firebase platform has 18 products. Firebase provides an alternative to server technology accumulating features separated as different products like authentication, database, storage, functions, hosting, etc. providing developers a painless platform to implement in their projects

In our project we have used a service of firebase called FireStore. Firestore is used as application's database

Describe your implementation briefly (best with sample code snippets) (1pt)

- First we installed firebase tools using npm
- Then we could login to firebase and connect the local created project to a firebase project created in firebase console

- We installed a firebase-functions-helper package

  https://www.npmjs.com/package/firebase-functions-helper

- The project was done in TypeScript
- We created a CRUD route for API
- We edited firebase.json for proper base routing of API
- We deployed the project to Firebase functions

index.ts

```
import * as functions from 'firebase-functions';

import * as admin from 'firebase-admin';
import * as firebaseHelper from 'firebase-functions-helper';
import * as express from 'express';
import * as bodyParser from "body-parser";

admin.initializeApp(functions.config().firebase);

const db = admin.firestore();

const app = express();
const main = express();

const contactsCollection = 'contacts';

main.use('/api/v1', app);
main.use(bodyParser.json());
main.use(bodyParser.urlencoded({ extended: false }));

// webApi is your functions name, and you will pass main as
// a parameter
export const webApi = functions.https.onRequest(main);
```

API for creating new contacts

```
// Add new contact
app.post('/contacts', (req, res) => {
    firebaseHelper.firestore
        .createNewDocument(db, contactsCollection, req.body);
    res.send('Create a new contact');
})
```

API for viewing all contacts

```
// View all contacts
app.get('/contacts', (req, res) => {
    firebaseHelper.firestore
        .backup(db, contactsCollection)
        .then(data => res.status(200).send(data))
})
```

package.json(defines dependencies)

```
{
  "name": "functions",
  "scripts": {
    "lint": "tslint --project tsconfig.json",
    "build": "tsc",
    "serve": "npm run build && firebase serve --only functions",
    "shell": "npm run build && firebase functions:shell",
    "start": "npm run shell",
    "deploy": "firebase deploy --only functions",
    "logs": "firebase functions:log"
  },
  "main": "lib/index.js",
 "dependencies": {
    "body-parser": "^1.18.3",
    "express": "^4.16.3",
    "firebase-admin": "~5.12.1",
    "firebase-functions": "^1.0.3",
    "firebase-functions-helper": "^0.5.8"
},
  "devDependencies": {
    "tslint": "^5.12.0",
    "typescript": "^3.2.2"
  },
  "private": true
}
```

firebase.json

```
{
  "firestore": {
    "rules": "firestore.rules",
    "indexes": "firestore.indexes.json"
  },
  "functions": {
    "predeploy": [
      "npm --prefix ./functions/ run lint",
      "npm --prefix ./functions/ run build"
    ]
  },
  "hosting": {
    "public": "public",
    "ignore": [
      "firebase.json",
      "**/.*",
      "**/node_modules/**"
    ],
    "rewrites": [
      {
        "source": "/api/v1/**",
        "function": "webApi"
      }
    ]
  }
}
```

After creating the microservice, we accessed the microservice through API from our web application built in JAVA.

Sending post request to API

```
String query_url = "https://sad3-e5b09.firebaseapp.com/api/v1/contacts";
URL url = new URL(query_url);
HttpURLConnection conn = (HttpURLConnection) url.openConnection();

conn.setInstanceFollowRedirects(false);
conn.setConnectTimeout(5000);
conn.setRequestProperty("Content-Type", "application/json; charset=UTF-8");
conn.setDoOutput(true);
conn.setDoInput(true);
conn.setRequestMethod("POST");
OutputStream os = conn.getOutputStream();
os.write(json.getBytes("UTF-8"));
os.close();

int responseCode = conn.getResponseCode();

if (responseCode == HttpURLConnection.HTTP_OK) {

    BufferedReader in = new BufferedReader(new InputStreamReader(

            conn.getInputStream()));
```

Using get request to retrieve data:

```
URL url = new URL("https://sad3-e5b09.firebaseapp.com/api/v1/contacts");
HttpURLConnection conn = (HttpURLConnection) url.openConnection();
conn.setRequestMethod("GET");
conn.setRequestProperty("Accept", "application/json");

conn.connect();

BufferedReader br = new BufferedReader(new InputStreamReader(conn.getInputStream(), "UTF-8"));

StringBuilder sb = new StringBuilder(2048);
for (String line; (line = br.readLine()) != null;) {
    System.out.println(line);
    sb.append(line);
}

JSONObject o = new JSONObject(sb.toString());
```

A diagrams/graphs explaining the architecture of the whole app (1pt)

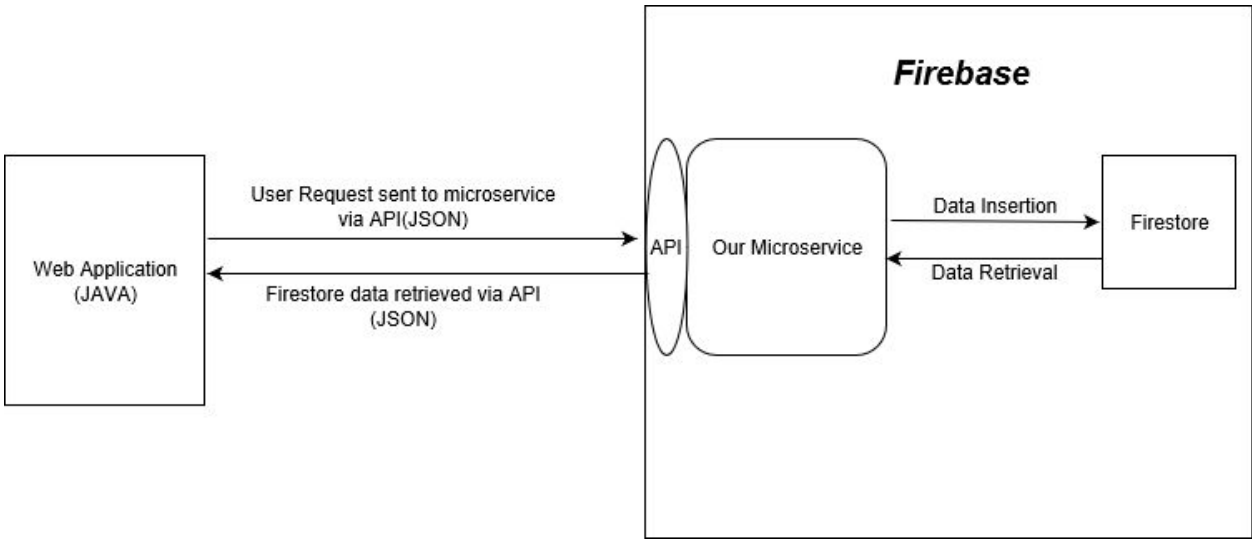

Fig: Project Architecture

POST method:

HTTP request is initialized from JAVA application to the API created by microservices. Data in API is provided in JSON format. The microservice then redirects the data to Firestore where the data is stored.

GET method:

HTTP request from JAVA application triggers the API for GET method and data from firestore is fetched and provided back to the application in JSON format. The application then interprets the JSON data and displays it in the interface.

Discuss the pros/cons of microservice and serverless technology in your own words

Pros:

- If a certain part of application needs to be changed, it be independently developed and deployed.
- Each microservice can be developed on different platform, using different programming language and different developer tool.
- Microservice is scalable and it promotes agility.
- Integration with 3rd party services is easy.
- Microservices has isolated services which lead to better fault tolerance

Cons:

- The complexity of microservice architecture makes it harder to test and monitor.
- Communication between microservices leads to poorer performance, as sending messages back and forth comes with a certain overhead.
- Unsafe distributed communication can occur as microservices run in a separated processes and communicate over the network.

References:

https://raygun.com/blog/what-are-microservices/

https://medium.com/@goodrebels/to-go-or-not-to-go-micro-the-pros-and-cons-of-microservices-7967418ff06

https://phauer.com/2015/microservices-nutshell-pros-cons/

https://smartbear.com/solutions/microservices/

https://hackernoon.com/what-is-serverless-architecture-what-are-its-pros-and-cons-cc4b804022e9?gi=3fd7c1b00013