Overview of Backend as a Service platforms

Report

Siim Plangi
Institute of Computer Science
University of Tartu
Tartu, Estonia
splangi@ut.ee

***Abstract*—Mobile application development is on the rise. This increases the demand to deliver mobile applications as fast as possible to put the minimum viable product to the test. One way to cut development time for mobile application is to minimize the backend development. This is where Backend as a Service platforms come to help, allowing to develop applications without any or little server side code. In the following paper, the author describes what Backend as a Service is, what its advantages are, what are its disadvantages and finally one platform will be analyzed closer using an Android application.**

Keywords—Backend as a service, BaaS, Backend, Mobile applications

I. INTRODUCTION

The demand for mobile applications is increasing every year and as it is said in business, time is money and therefore apps need to be tested as soon as possible. Therefore there is a demand to develop and deliver mobile applications faster than ever before. One way to do that is skip backend development altogether and focus only on front-end development. This can be achieved with the help of Backend as a Services (BaaS), by almost eliminating any need to write server side code. This allows faster development and quicker MVP (Most valuable product) releases which can be used to test out business ideas. This is the reason why BaaS has become increasingly popular over few years.

This paper makes an introduction to Backend service products. The advantages and disadvantages of BaaS will be analyzed and one product will be overviewed closer using an Android application to demonstrate the functionality and possibilities.

II. BACKEND AS A SERVICE

Developing a fully featured mobile application often depends on server side capabilities. To lessen the burden of developing server side code, a new Platform as a Service type has emerged: Backend as a Service. BaaS gives to applications the most common Server side capabilities, like create, read, update, delete (CRUD) operations for data and authorization, in a cloud-hosted model. Usually BaaS does not require any server side code writing and is fairly easy to setup thus reducing development time.

Nowadays, as the mobile application world is booming – demand is increasing day by day as well as Baas providers. Some examples are:

- Firebase
- Kinvey
- Parse
- Appcelerator Cloud
- Backendless

A comprehensive list of over 50 service providers can be found in GitHub project named "ParseAlternatives" [1].

All these providers differ slightly and offer their own advantages over others but most of them have these key features [2].

- CRUD operations – The ability to create, read, update and delete data.
- Authorization – integration with existing social OAuth providers.
- Native notifications for mobile applications – Sending push notifications to engage or notify users.
- Query functions over data – Search and retrieve specific parts of data to reduce network latency and memory consumption.
- Administration console – Edit the configuration of the BaaS. May include a visual data representation with data editing options

Other aspect what most of the BaaS providers have in common is that for data storage they mostly use NoSQL. This is slightly different data storage way than traditional SQL databases. Instead of relational structured database schema, NoSQL uses non structured data saving, usually in the form of one big JSON file [3].

III. Benefits of BaaS

In the following paragraphs the advantages of BaaS will be described [4, 5, 6].

A. Fast development

Backend as a Service platforms allows developing mobile applications with remote data storage and processing, whilst writing little or no server code at all. This in turn means, that valuable time and money is saved while developing the application, which can be spent in other areas of application development.

B. Cheap

Most BaaS services offer entry level services cheaply or free. This again helps the development team to test out their app idea with as little budget as possible. Also, if the main goal is not business but a personal or educational project, then this helps to keep the overall cost to a minimum. There are even open source services which one can setup in their own server, making sure that the costs of using the service will be low and controlled in the future.

C. Easy setup

Setting up a BaaS platform is usually very easy. The developer usually needs an account in the service provider website, make some configurations and the BaaS is ready to be used. If there are some software development kits (SDK) additional setup might be needed for each individual SDK. For open source BaaS, setup might be a bit more complicated because of setting up the servers and installing dependencies, but usually this is made quite easy and clear in the setup tutorials.

D. Scaleable

BaaS platforms are built to service any kind of application and must be very flexible. Therefore, small applications which go viral, usually will not be affected by performance loss, because of automatic cloud scaling. Usually this means paying more money, but this is much less troublesome than setting up multiple servers yourself.

E. Additional features

In addition to saving and retrieving data, many of the BaaS engines offer additional features to differ from one another. Therefore developers can find the solution for their needs. Most notable additional features are:

- SDKs – BaaS platforms offer various client side software development kits to even further increase the ease of development in the same language as the client code.
- Real-time database – All the changes in data are broadcasted automatically to all listeners.
- Push notifications – The possibility to send notifications to mobile applications to increase engagement.
- Server side code – Some of the BaaS engines offer to write limited server side logic.

F. Many platforms to choose from

As written before in this paper, there are many different services to choose from. This means that developers can find the best tool to use for their needs. When choosing a Backend as a Service, one should consider the following:

- Pricing
- Scalability
- Additional features
- Security

IV. Disadvantages of BaaS

In the following paragraph, the disadvantages of BaaS will be discussed [4, 5].

A. Lots of competition

There are a lot of different choices for BaaS platforms. On one hand, this gives the possibility to choose the right platform for the mobile application. On the other hand, this large competition means that the service provider might be closed or be acquired by some other company which might bring complications.

This happened with one of the most popular BaaS engines Parse. Parse was acquired by Facebook in 2013, which meant that the platforms future and APIs were in control of Facebook [7]. In January 2016 it was announced that the Parse engine will be discontinued with closure on 28. January 2017 [8]. This demonstrates the two main problems discussed.

B. Does not support complex backend logic

Some of the BaaS platforms (aka BaasBox [9]) offer writing server side code, but this is fairly limited. For more complex logic one still needs a conventional backend between the BaaS platform and application, which then can take care of more complex backend logic.

C. Pricing

Although most of the BaaS platforms can be considered cheap at entry level, when the application grows the pricing can increase rapidly as well, making the BaaS less cost effective. Therefore, when choosing a BaaS solution, one has to consider the scenario of application growth and choose the platform accordingly.

V. FIREBASE

Firebase is one of the many BaaS offerings available. It differs from the other BaaS services being a real time backend. This means that updates to the data are immediately broadcasted to any client that is listening for updates. Firebase is really easy to set up and offers various SDKs to start developing as easy as possible. It also has a great documentation for using all the different SDKs. This BaaS platform was chosen because the author of this document wishes to use this service in his future projects.

To learn and demonstrate different capabilities of BaaS and Firebase a small Android project was created. This was a simple maps application that allows anyone to broadcast their location to the server. Anyone who is listening for location updates can see live positions of others. There is also login capability using Facebook login feature offered in Firebase to learn about security and authorization features. Figure 1 demonstrates the main view of the developed application as well as two live locations of two different android devices. The source code has been uploaded to GitHub [10].

Figure 1 - Test android project

The following paragraphs demonstrate the usage of Firebase, all of the info has been taken from the Firebase documentation [11] and from the knowledge obtained from developing the test project.

A. Data structure

Firebase uses the NoSQL data storage format which most other BaaS solutions use. All objects saved into and read from databases are JSON objects. This means that the database is one big JSON file which has arbitrary structure.

```
{
    "users": {
        "41fc01e3-0082-4c78-8ce0-57c081089323": {
            "userLocation": {
                "lat": 58.39001,
                "lng": 26.72739
            }
        },
        "facebook:1088066984582757": {
            "name": "Siim Plangi",
            "userLocation": {
                "lat": 58.3823411,
                "lng": 26.7173781
            }
        }
    }
}
```

Figure 2 - Example data from Firebase

In Java, the JSON data is translated in to the following objects:

- String
- Boolean
- Long
- Double
- Map<String, Object>
- List<Object>

This is a recursive list, where all the Object class fields will be serialized to the same types as in the given list if possible.

B. Data entry

Entering data into Firebase using the Android SDK is quite straightforward. Firstly, one needs a reference path. This path is composed of the applications database address and path to where the new data will be saved. There are four different ways to save data to server:

- setValue() method – Writes or replaces all the data in the specified path
- updateChilder() method – Updates data with updated fields rather than replacing all the data.
- push() method – Adds to a list of data. Push() method generates a unique ID for this object.
- runTransaction() method – Used when concurrent modifications of data could corrupt the data, such as number incrementation.

C. *Data retrieval and updates*

Data retrieval and updates are handled attaching listeners on the Firebase reference objects. There are two different event listeners available in Firebase Android SDK.

- ValueEventListener – Used to read the whole data and listen changes for the whole data on that specific path. Every time any data changes appear in that node or within its children nodes whole data is returned
- ChildEventListener – Used when usually dealing with lists of data. When the reference path child changes then one of the following methods is triggered.
    - onChildAdded – When a child data is added to the reference path
    - onChildChanged – When a child data is changed in the reference path
    - onChildRemoved – When a child data is removed from the reference path
    - onChildMoved – When the children are ordered and this data has changed its position

D. *Security*

Firebase offers a powerful security mechanism to secure data from unwanted Create-Read-Update-Delete actions. Firebase uses Security rules to define different access levels to users on different datapaths. Figure 3 shows an example of a rules file, which gives full read/write access to everyone on any data path.

```
{
    "rules": {
        ".read": true,
        ".write": true
    }
}
```

Figure 3 - Example of Security rules file

There are two different techniques to protect data.

1) *Authorization*

Authorization security rule provides a way to check if the user writing/reading data is who it claims to be. For this Firebase offers the auth and auth.uid variable. Auth variable is not null only if the user has authenticated using Firebase and auth.uid is the user's id. Figure 4 shows a security rule, that disallows write to users datapath if the current auth.uid is not equal with the $user_id variable.

```
{
  "rules": {
    "users": {
      "$user_id": {
        ".write": "$user_id === auth.uid"
      }
    }
  }
}
```

Figure 4 - Example of security rule

2) *Data validation*

Data validation is used to check if data is entered correctly. Figure 5 shows an example that checks if entered coordinates are within the range of real coordinates.

```
{
    "rules": {
        "users": {
            "userLocation": {
              ".validate": "newData.child('lat').val() < 90
              && newData.child('lat').val() > -90
              && newData.child('lng').val() < 180
              && newData.child('lng').val() > 180"
            }

        }
    }
}
```

Figure 5 - Example of security rule

When both of these techniques are combined then the security mechanism can be complex and very powerful.

E. *Users and Login*

Firebase provides 7 different login mechanisms. Four of those are Social media logins.

- Facebook
- Twitter
- Github
- Google

Three other login ways are:

- Email & Password
- Anonymous login
- Custom login using JSON Web Tokens (JWTs)

These authorization methods should cover the needs of most applications.

VI. Conclusions

To conclude, using BaaS platforms is a great way to reduce development time and resources. As seen, the usage of Firebase was straightforward and a relatively complex application was made in a fraction of the time, that it would have taken with traditional backend development. If the choice of BaaS is done right from the vast amount of different platforms then it can eliminate the need for writing any server side code, have necessary server side functions for the applications and be low cost or even free. But one needs to take care when choosing the right BaaS, as there are dangers as well. Firstly, as there are so many choices then there is the risk of platform shutdown. Secondly, the pricing can be seductive and deceptive at entry level, but when the application scales up, the pricing might scale up as well. Thirdly, when there is a need for complex server logic, there is no escape from traditional backend development.

VII. References

1 "GitHub," [Online]. Available: https://github.com/relatedcode/ParseAlternatives. [Accessed 4 4 2016].

2 M. Facemire, "Mobile Backend-As-A-Service: The New Lightweight Middleware? Michael Facemire," 25 April 2012. [Online]. Available: http://blogs.forrester.com/michael_facemire/12-04-25-mobile_backend_as_a_service_the_new_lightweight_middleware. [Accessed 4 04 2016].

3 R. Cattell, "Scalable SQL and NoSQL Data Stores," 12 2011. [Online]. Available: http://www.cattell.net/datastores/Datastores.pdf. [Accessed 04 04 2016].

4 G. V. Huizen, Interviewee, *Backend as a Service: Reap the benefits, master the challenges.* [Interview]. 08 2013.

5 P. Shetti, "What is Baas? And Benefits of BaaS Solution," 26 04 2016. [Online]. Available: https://www.wattpad.com/181041496-advantages-and-disadvantages-of-backend-as-a. [Accessed 2016 04 26].

6 C. Bedell, "Mobile backend as a service: Benefits for enterprise agility," Mobile Business Insights, 30 11 2015. [Online]. Available: http://mobilebusinessinsights.com/2015/11/mobile-backend-as-a-service-benefits-for-enterprise-agility/. [Accessed 26 04 2016].

7 J. C. Kim-Mai Cutler, "TechCrunch," 25 04 2013. [Online]. Available: http://techcrunch.com/2013/04/25/facebook-parse/. [Accessed 25 04 2016].

8 K. Lacker, "Parse Blog," Parse, 28 01 2016. [Online]. Available: http://blog.parse.com/announcements/moving-on/. [Accessed 25 04 2016].

9 BaasBox, "BaasBox Blog," BaasBox, 17 09 2014. [Online]. Available: http://www.baasbox.com/beyond-the-box-we-are-launchingcustom-server-coding-with-baasbox/. [Accessed 25 04 2016].

10 S. Plangi, "Github," [Online]. Available: https://github.com/splangi/DS_Firebase_Test. [Accessed 26 04 2016].

11 Firebase, "Firebase documentation," [Online]. Available: https://www.firebase.com/docs/. [Accessed 26 04 2016].

12 W. Kohl, "Backend As A Service, Using the Example of Enginio a Cloud Service," 2014. [Online]. Available: https://www.uni-ulm.de/fileadmin/website_uni_ulm/iui.inst.200/files/staff/K%C3%A4chele/kohl14backend-as-a-service-enginio.pdf. [Accessed 2016 04 26].