Conference Room Reservation Application

HAMK
HÄMEEN AMMATTIKORKEAKOULU
HÄME UNIVERSITY OF APPLIED SCIENCES

Bachelor thesis

Degree Programme in Business Information Technology
or Computer Applications
Hämeenlinna University Centre
Spring, 2021

Ward Huysmans

HAMK
HÄMEEN AMMATTIKORKEAKOULU
HÄME UNIVERSITY OF APPLIED SCIENCES

Degree Programme in Business Information Technology
Hämeenlinna University Centre

Abstract

Author Ward Huysmans
Subject conference room reservation application
Supervisors Deepak KC

Year 2021

ABSTRACT

The purpose of the thesis was to create a proof of concept for the commissioning party. This solution is for an application that will be used for booking a waiting room. This includes creating the web API, creating the application for the tablets in front of the waiting rooms, the communication and security between the application and the web API. The commissioning party is a company in Belgium called Tobania. It is a business and technology consulting company.

To create this proof of concept, many studies, videos, and blogs have been used. The results of these researches have been combined to form an argument for each option, which then was discussed with the supervisor. After the discussion, the solution was then added to the proof of concept.

This resulted in an acceptable proof of concept which can later be used in the company, to create this application.

Keywords architecture, communication, SDK, security, hardware

Pages 27 pages and appendices 1 page

Glossary

SDK	Software development kit
API	Application programming interface
UWP	Universal Windows Platform
Pub/Sub	Publish/Subscribe
MVVM	Model View ViewModel
XAML	eXtensible Application Markup Language

Contents

Annexes

Annex 1 Material management plan

1 Introduction

This thesis aims to create a proof of concept which can be used in future development. The result should be easy to read and use.

Tobania is the leading Belgian Business & Technology Consulting firm, Tobania guides companies through their digital business transformation. Their professional employees will help other companies with their transformation programs.

The topic was suggested by Anton Kirschhock who works for Tobania as a developer. Anton asked around in the company if there were any projects suitable for the author. Finally, three proposal projects were set. The first proposal was this project which was suggested by the IT department. The second project was to update the user interface for one of their applications. The last project was to create a virtual reality application that would show the inside of Tobania so clients can walk around and see it for themself. This was suggested by the marketing department. The first option was chosen, this was the most appealing project to me because it was a combination of different aspects of IT.

The following research questions were set:

How is the security going to be between the tablet and the web application programming interface (API)?

How is the communication going to work?

What is the architecture going to look like?

2 The proof of concept

A proof of concept is an exercise to check the design plan or assumption. The main objective of creating a proof of concept is to demonstrate the functionality and to verify an explicit concept or theory that may be achieved in development. In order to form the proof of concept, everything ought to be looked at step by step. The big questions should be what the project requires and what the options are to reach these goals. (Singaram Muthu, Prathistha jain, 2018)

2.1 Hardware

2.1.1 Raspberry Pi

A big plus is that it is easy to repair and maintain since the screens need to be built by the company. The components will be cheaper than the components for Android tablets. This solution would also be more beneficial to distribute this to other companies. A Raspberry Pi tablet would also be better if Tobania wants to expand the functions of the tablet.

Besides this, Tobania does not own any screens powered by Raspberry Pi. This would mean the company has to order the screens and additional components. It will also take more man-hours and Tobanie would need someone who specializes in Raspberry Pi projects. This solution would be overkill for the project.

2.1.2 Android tablet

At the moment, the company has Android tablets in stock, so cost-related this would be the best solution. An Android tablet would also satisfy the needs of the customer. It is also very easy to work with. When there is a problem with the hardware of the tablet, it is more difficult to repair it. The tablets already have a batch reader attached to them so there is so no extra materials need to be bought. Android tablets are cost-effective and were considered satisfactory for the client/customer.

2.2 Software development kit

2.2.1 Flutter

Created by Google, Flutter is a cross-platform open-source software development kit (SDK) to build apps for iOS & Android. Apps and interfaces created with flutter are built from a single codebase, compiled onto native arm code, use the GPU, and might access platform APIs and services. Even though it is the newest SDK, Flutter had become a favourite for developers. (Jelvix, 2020; Von Der Howen Georg, 2020)

The author has no experience with flutter. After working with the SDK, the author decided that this will not be used for the project.

Experience

Visual Studio Code was used as the source-code editor for this project. There were a lot of problems setting everything up. There needed to a lot of things download before it could even start, which took a lot of time, and time is very precious for a developer. When it comes to using Flutter there were no major complaints. There had to be some research done from time to time on how this environment works but that is very common with a new tool. The result worked but there were some bugs that the author could not solve in the given time. Overall It was a good first try with an uneven start.

Pros

- Flutter supports hot reload, which works by injecting updated source code files in the running Dart Virtual Machine.
- Flutter permits building native interfaces without preparing two apps.
- All elements have a clear hierarchical structure.

Cons

- While dart is inspired by JavaScript, it is still a new language.

- If there is a need to build a complex interface the performance will drop instantly.

- If the app's activity stops Flutter does not conserve the data of it anymore.

2.2.2 Xamarin

Xamarin is Microsoft's technology for creating native mobile applications. It can be used for Android applications, iOS applications, and Universal Windows Platform (UWP) applications. It uses .NET and C# to create these apps. For this project, the decision was made to use Xamarin.Forms. It uses Model View ViewModel (MVVM) which is a design pattern. (AltexSoft, 2018)

Because of the author's experience with MVVM from previous projects, no problems occurred when using it. The author's preference would go to Xamarin. A big factor in the decision was also the author's prior knowledge of C# and eXtensible Application Markup Language (XAML) which will make the programming part easier and faster.

Experience

Visual Studio was used as the source-code editor for this project. Since Xamarin comes with Visual Studio it only needed to be downloaded and everything was ready. Because of the author's experience, the application was created fast. The result was that the application was less advanced than the Flutter one. This was mostly because of time. Overall it was a good experience.

Pros

- There are only three components needed to create apps in Xamarin. There needs to be knowledge about C#, .NET and the computer needs to have Visual Studio with Xamarin installed.

- Xamarin apps have a better performance level compared to other cross-platforms SDK's.

- It is open source.

Cons

- There are always delays with updates.
- The size of the applications is going to be bigger since the whole .NET runtime is included.

2.2.3 .NET MAUI

.NET MAUI is an SDK that is used for building native device applications on mobile, tablet and desktop. It is the evolution of Xamarin. Since the release date is scheduled for November 2021, this could not be tested. There is also not enough known about this SDK. It will probably still have many bugs in the beginning. Sadly enough it can not be considered useful for this project, but maybe for a future version.

2.3 Communication

2.3.1 Real-time data processing

Real-time data processing is the implementation of data in a short period, providing near-instantaneous output. The processing is finished because the data is inputted, therefore it desires a continuous stream of input data to supply a continuous output. (Wilson Christy, 2020) Real-time data processing is going to be very heavy on the application and there is a possibility it can not handle that.

Pros

- There is going to be very low latency.
- With real-time data, there is a possibility for interventions.

Cons

- There is no possibility to summarize data because it works with very small time windows.
- The data will not be persisted for deeper analysis.
- It can be tough to add any type of even moderately complex calculation logic.
- The user cannot immediately see that the task has been completed.

2.3.2 Near real-time data processing

Near real-time advert to data processing and communications that quickly reply to events shortly after it occurs. in contrast to real-time processing, near real-time implies that processing isn't optimized to be as quick as attainable. The time concerned in near real-time processing depends on the problem space. A delay of minutes, seconds, or milliseconds is often considered near real-time. (Wilson Christy, 2020; Pluster Matt, 2019)

Near real-time data processing is also a very close candidate, but not knowing when data is going to be processed is too much of a risk.

Pros

- The ability to persist data is a big advantage. This means that there is a possibility to store the data somewhere that is not coming directly off the stream.
- Near real-time data can look at large windows of time to do historical and cross-domain analysis
- It still has a very tight time window as far as how the data is transferred.

Cons

- There can be latencies that can take 5 to 15 minutes or sometimes even longer. This is because of the need to first persist the data and then process it.

- It is not possible to do an immediate intervention, this means that the data can only be evaluated after an event has occurred.

2.3.3 Batch processing

With batch operations, the source data is loaded into data storage, either by the source application itself or by the orchestration workflow. The data is then processed in place by a parallelized job, which might also be initiated by the orchestration workflow. The processing may include multiple iterative steps before the reworked results are loaded into an analytical data store, which might be queried by analytics and reporting components. (Microsoft, 2018)

The author thinks that the communication between the application and the API will take too much time which would be a big problem.

Pros

- Everything is done in one single process.

Cons

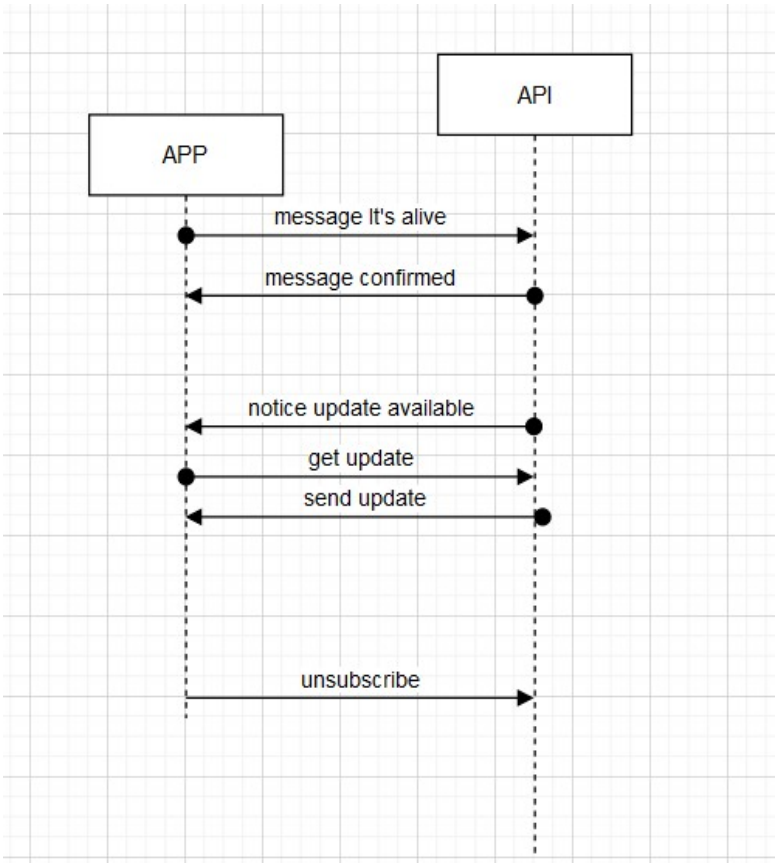
- The data takes time to be processed.

2.3.4 Pub/Sub messaging

The Publish/Subscribe pattern is an architectural design pattern that delivers a framework for swapping messages between publishers and subscribers. This pattern involves the publisher and also the subscriber relying on a message broker that relays messages from the publisher to the subscribers. The host publishes messages to a channel that subscribers can then sign up to. The author's thought is that for the project this the most interesting one is. (Google Cloud, 2019)

The author thinks this will be the best solution for the application.

Figure 1: Pub/Sub messaging scheme



How this works is the moment the application is live, it will send a message to the API saying that it is live and that it wants some updates. The API will then in return send a notification that it got the message and add the application to the client list. The API will afterward send a message when the update is available. The application will then try to receive the data with an HTTP request and the API will send it. The API can also send the update immediately. At the end of the cycle when the application will shut down, it will unsubscribe itself from the API. The API will remove the application from the list of clients.

Pros

- The publisher and subscriber work independently. This means that both of them can be developed separately without worrying about the state or the implementation.

- It can be easily figured out if a publisher or subscriber is getting the wrong messages.

- Pub/Sub allows the systems to scale, however it buckles under load.

Cons

- Intruders can invade the system and breach it, this can lead to bad messages being published and subscribers having access to messages that subscribers should not receive

- As the number of subscribers and publishers increases, the increasing number of messages being exchanged leads to instabilities in this architecture.

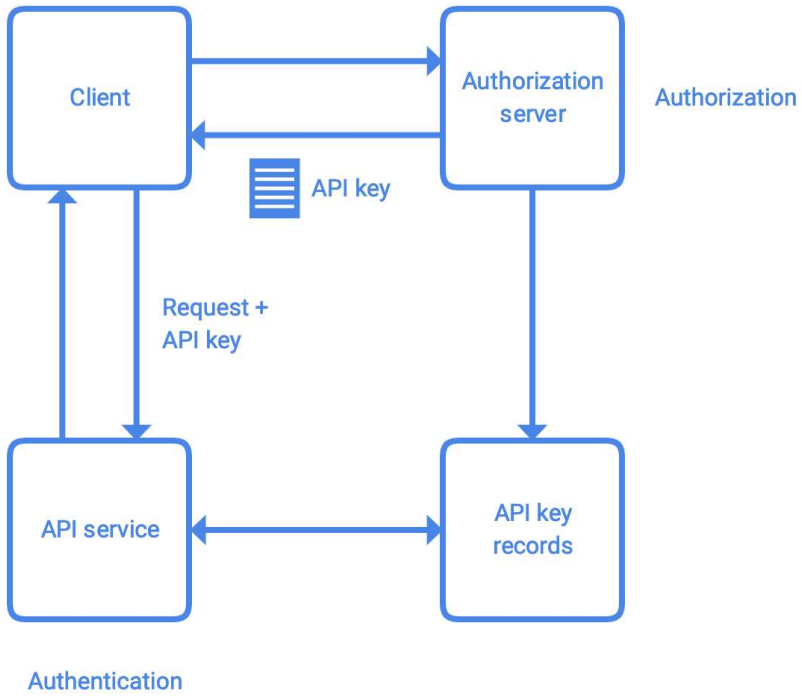
2.4 Security

2.4.1 API key

An API key is a randomly generated string prepared in a cryptographically strong manner. It is one of the most famous types of security. Two types of keys are used to access the search service: admin (read-write) and query (read-only). The admin grants full rights to all operations, including the ability to manage the service, create and delete indexes, indexers, and data sources. The query grants read-only access to indexes and documents, and are typically distributed to client applications that issue search requests. (Addie Scott, Dykstra Tom, 2020)

In the author's eyes, API keys should be used when developers want to build an internal application that does not need to access more than a single user’s data, which is not the case in the project. The author thinks that an API key would work but there is a better option for the project.

Figure 2: API key scheme (Ratios Y,2019)



As shown in figure 2, the client, which in this case is the application, requests an API key from the authorization server. The authorization server gives one to the client and adds the application to the API key records. The application is going to request data from the API with the API key. The API checks if it can find the key in the records. If the answer is yes, then it is going to send the data to the application. (Ratros, 2019)

Pros

- API keys are straightforward to integrate when using an API management solution.
- API keys are exceptional at limiting the chance of read-only data.

Cons

- Read-only API keys are limiting once it comes to data that requires specific permissions.
- While API keys excel in authenticating users, it struggles with authorizations.

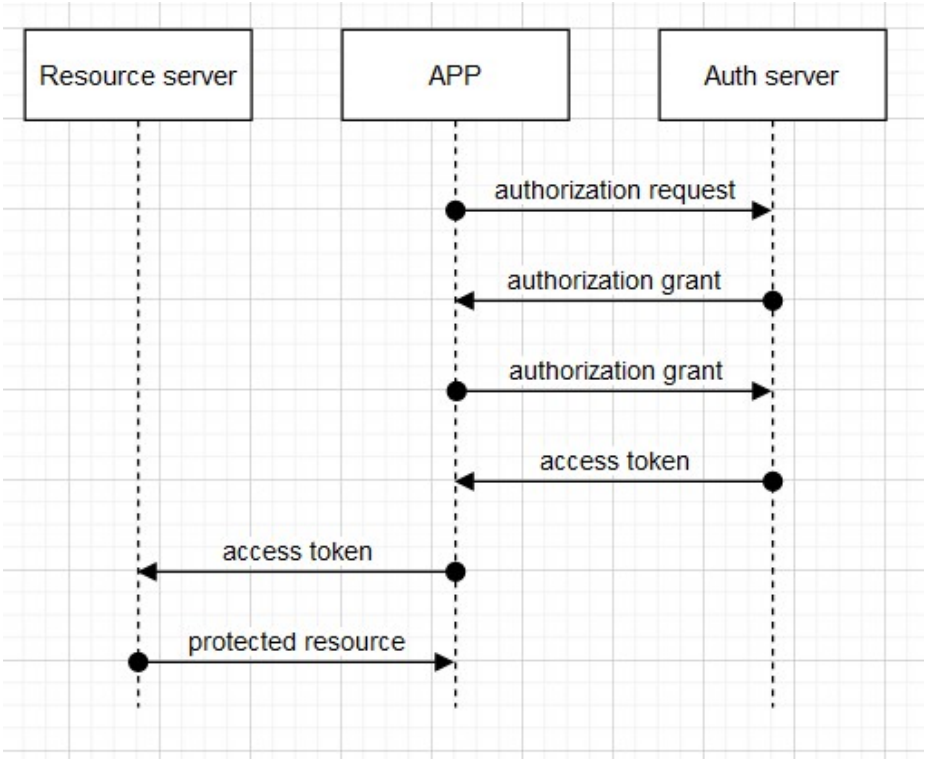
- A broken API key can annihilate data.

2.4.2 OAuth

OAuth 2.0 is a standard protocol for authorization. It focuses on client developer simplicity while contributing particular authorization flows for web applications, desktop applications, mobile phones and living room devices. The OAuth 2.0 authorization code grant can be utilized in apps that are installed on a device to gain access to protected resources, like web APIs. (Microsoft, 2021)

The author's thought is that OAuth 2.0 is the best solution for the project. It provides authorization to applications without needing to share private data.

Figure 3: OAuth 2 scheme



First, the application is going to request the authorization server. The authorization server asks the person in question if it is ok that the application accesses the resources that are needed to run the application. The authorization server sends the accepted authorization grant along with an

authorization code that the application can use to request an access token to see data from the person. The application is going to use the authorization grant along with the authorization code and uses it to request an access token. After accepting the authorization grant and code, the authorization server provides the application with an access token specifically for this user. The token will be included in the request from the application to the resource server. The resource server identifies that this token is valid to access only the resources needed to run the application. It sends the protected resources back to the application and now the application has the resources needed to get the application running.

Pros

- OAuth security tokens are tremendous at authorizing levels of access for specific users.
- OAuth security tokens may be set to expire.
- OAuth security tokens provide exceptional access to user data.

Cons

- OAuth security is a lot more complicated than API key security.

3 Methodology

3.1 methodological approach

The first objective is to find sources that can help decide which topics the application needs. Just to get a starting point every time there is a new subject, there were meetings held with the client for possible options so that there is a starting point.

3.2 methods of data collection

The best method for the author was to read papers and blogs. There is also the possibility to watch videos, which are better for the author to understand sometimes. In the case of the SDK, the options have to be tested manually by the author. This is the best way to determine since most of the time a programmer has to choose the one which suits his taste best.

3.3 methods of analysis

For the thesis, the goal was to use a lot of different sources for the subjects, since everyone has a different view on certain topics and some people can see flaws that others cannot. The goal was to try to get 3 sources for each option. In the end, after a reading session, the author chose the one which had the better arguments. Not to say there were no materials from other sources used. The strategy was to check the website from the possible solution first and look at which features the website said were their strong points. Later the plan was to compare those with research papers and blogs about that option.

3.4 justify the methodological choices

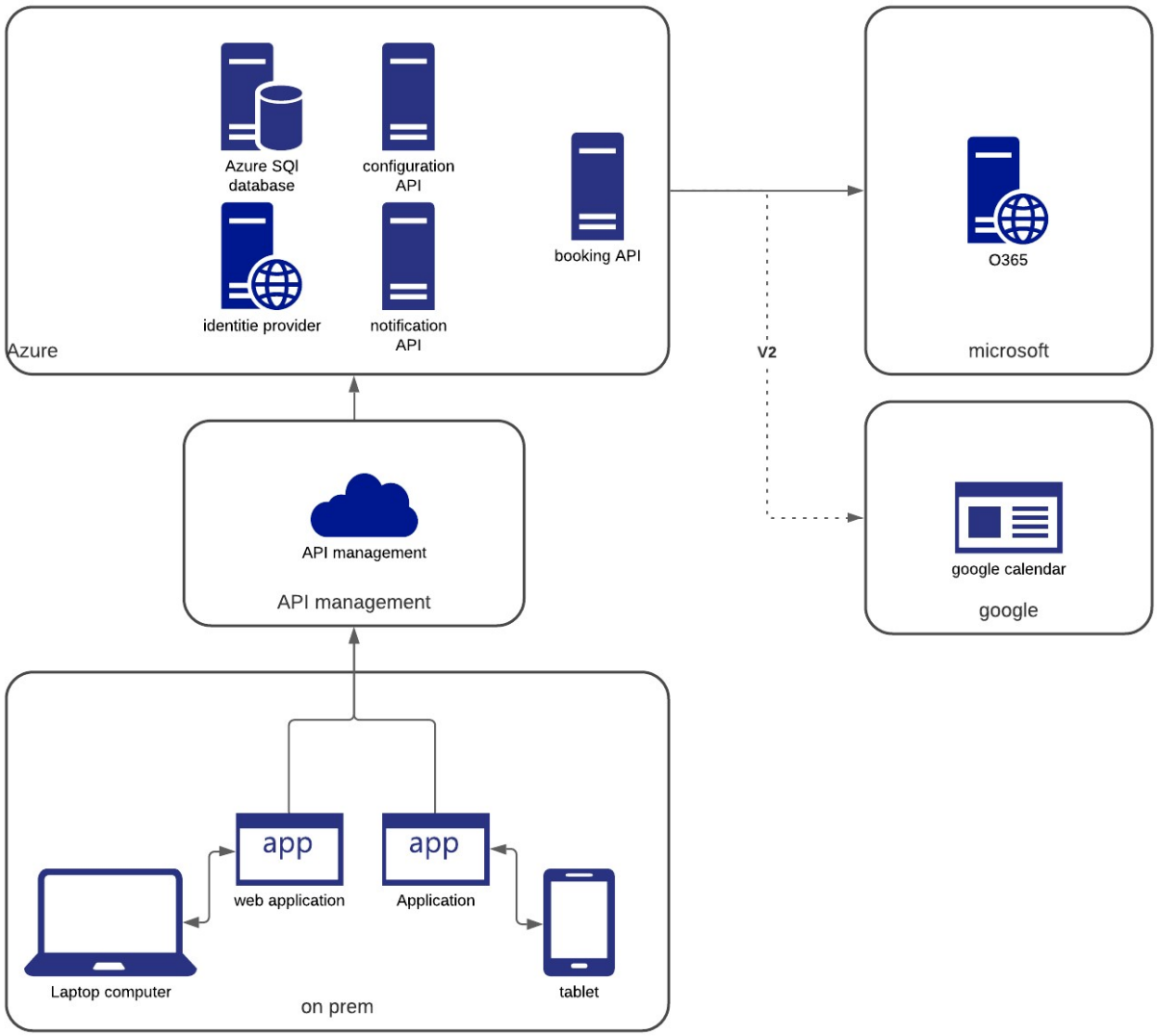
In the author's eyes, the research was successful, and that the choices that were made were the right ones for the proof of concept. There are probably some things that the client wants the author to investigate further since technology evolves and something new is always worth a look at.

4 Architecture

Let us now look at how the build of the project is going to look like. An application architecture specifies the patterns and techniques accustomed to design and build an application. The architecture gives the author a roadmap and best practices to follow when building an application so that the project ends up with a well-structured app.

4.1 Application Architecture Diagram

Figure 3: Application Architecture Diagram



4.2 Architecture diagram detailed

4.2.1 On-prem layer

This layer is about everything the developer can see and interact with. It contains four components: the laptop/computer, the tablets, the web application, and the application on the tablets. One can see the web application and the application both communicate with the API management layer.

4.2.2 API management layer

This layer is the gateway between the application and the APIs. The API management is going to take all the different APIs and put them all together as one big API. Because of this, the tablet will not know anything between the different APIs. It only knows that that is the endpoint and it has to communicate with it.

4.2.3 Azure layer

This layer contains five components: the booking API, the identity provider, the notification API, the configuration API, and the SQL database. The booking API is used for the booking of the meetings. It will communicate with other layers to create and get info from the meetings. The identity provider will be used for multitenancy. This means that it will provide the creator with which tenant the user is located. The notification API is going to give the application a notification when there is an update. The configuration API is going to give the configuration which calendar the booking API has to communicate with. The SQL database will save the data for which configuration belongs to which user.

4.2.4 Microsoft O365 layer

This layer contains the O365 server from Microsoft. The booking API is going to communicate with this server to create the meetings in teams. Since Tobania's main way of communication within the company is through teams, the author should leave that unchanged. The author thinks it is

best to not create a new meeting application but just use the one that the company already uses for the creating of the meetings.

4.2.5 Google layer

This layer contains the Google calendar application. Tobania will use this in a later version of the application but it is already good to put it in there for future reference. This layer talks with the booking API and every time there is a new booking the user's Google calendar will add this to the user's calendar.

5 Results

All the research questions have been answered. The first research question was how the security was going to be between the tablet and the web API. The answer to this was that the project is going to use OAUTH 2. This is because it provides authorization to applications without needing to share private data. The next question was how the communication is going to work. For this question, the answer was a Pub/Sub messaging communication. This was suggested by the boss as a very good solution to look into. It was true, the subscriber and publisher are not too heavy for application since the subscriber will request data when it gets a notification from the publisher that there has been a change. The last question was how is the architecture going to look like. This is answered in the architecture part. There is a sketch of how it all should look and a description of every component. We also decided that we are going to use android tablets instead of the raspberry PI tablets. The main reason being that the company already owns android tablets. The author also decided that it will use Xamarin as the SDK after testing the options.

5.1 Further activities

Now the project should be in the early development stage. During this stage, a group will be created to create and oversee this project. The author will be part of that group as one of the developers. The author thinks the group needs at least one more developer since the work experience that the author has is pretty low. The group should also contain someone with networking experience to set up and oversee the APIs.

6 Summary

The thesis was a great experience where the author learned a lot about how projects are set up in the business environment. The difficulty level for this project was well chosen by the client. It also covered a lot of different categories in IT. It contained some programming, some security and communication, and also some hardware. Researching all these different topics was not easy but it was educational. The author is relieved that the research questions are solved. The most difficult part for the author was the architecture. The author had no previous experience in creating these schemes. The author was fortunate that the client advised every step of the way to put the author on the right track.

The client was satisfied with the result. Anton told me that there are still some parts that need improvement, but Anton was satisfied with the result. The client told the author that they will soon start the project and they will use the thesis as their base. The client suggested that it would be good practice for the author to join the team and create the project with them. The author accepted this request.

7 Bibliography

Abhishek. 2018. “Layered Architecture with Azure API Management, Azure Functions, Azure Key Vault and Cosmos Graph… | by Abhishek | Medium.” Retrieved March 19, 2021 (https://medium.com/@abhishekcskumar/layered-architecture-with-azure-api-management-azure-functions-azure-key-vault-and-cosmos-graph-b120d98c9146).

Addie Scott, and Dykstra Tom. 2020. “Create Web APIs with ASP.NET Core | Microsoft Docs.” Retrieved January 25, 2021 (https://docs.microsoft.com/en-us/aspnet/core/web-api/?view=aspnetcore-5.0).

AltexSoft. 2018. “(30) Pros and Cons of Xamarin Development - YouTube.” Retrieved February 19, 2021 (https://www.youtube.com/watch?v=Yy29uRKpzhY).

Anderson Rick, Larkin Kirk, and Wasson Mike. 2020. “Tutorial: Create a Web API with ASP.NET Core | Microsoft Docs.” Retrieved January 25, 2021 (https://docs.microsoft.com/en-us/aspnet/core/tutorials/first-web-api?view=aspnetcore-5.0&tabs=visual-studio).

Anon. 2018a. “Batch Processing - Azure Architecture Center | Microsoft Docs.” Retrieved April 6, 2021 (https://docs.microsoft.com/en-us/azure/architecture/data-guide/big-data/batch-processing).

Anon. 2018b. “Publisher-Subscriber Patroon - Azure Architecture Center | Microsoft Docs.” Retrieved April 6, 2021 (https://docs.microsoft.com/nl-nl/azure/architecture/patterns/publisher-subscriber).

Anon. 2019a. “What Is Pub/Sub? | Cloud Pub/Sub Documentation | Google Cloud.” Retrieved March 8, 2021 (https://cloud.google.com/pubsub/docs/overview).

Anon. 2019b. “Why and When to Use API Keys | Cloud Endpoints with OpenAPI.” Retrieved January 26, 2021 (https://cloud.google.com/endpoints/docs/openapi/when-why-api-key).

Anon. 2021a. “API-Sleutel Verificatie - Azure Cognitive Search | Microsoft Docs.” Retrieved April 6, 2021 (https://docs.microsoft.com/nl-nl/azure/search/search-security-api-keys).

Anon. 2021b. “Microsoft Identity Platform and OAuth 2.0 Authorization Code Flow - Microsoft Identity Platform | Microsoft Docs.” Retrieved April 6, 2021 (https://docs.microsoft.com/en-us/azure/active-directory/develop/v2-oauth2-auth-code-flow#protocol-diagram).

Anon. n.d. “About - Tobania.” Retrieved March 12, 2021a (https://www.tobania.be/en-gb/About).

Anon. n.d. “Device Authorization Flow.” Retrieved February 9, 2021b (https://auth0.com/docs/flows/device-authorization-flow).

Anon. n.d. “Hot Reload - Flutter.” Retrieved March 24, 2021c (https://flutter.dev/docs/development/tools/hot-reload).

Anon. n.d. “What Is an Application Architecture?” Retrieved March 22, 2021d (https://www.redhat.com/en/topics/cloud-native-apps/what-is-an-application-architecture).

Anon. n.d. “What Is Real-Time Data Processing? - Definition from Techopedia.” Retrieved March 19, 2021e (https://www.techopedia.com/definition/31742/real-time-data-processing).

D. Hardt. 2012. “RFC 6749 - The OAuth 2.0 Authorization Framework.” Retrieved March 31, 2021 (https://tools.ietf.org/html/rfc6749).

Ehrli Erika. 2017. “Connect(); 2017: SmartHotel360 Demo Apps and Architecture | Visual Studio Blog.” Retrieved March 15, 2021 (https://devblogs.microsoft.com/visualstudio/connect-2017-smarthotel360-demo-apps-and-architecture/).

Gibb Robert. 2019. “What Is Pub/Sub Messaging? A Simple Explainer. - DEV Community.” Retrieved February 24, 2021 (https://dev.to/gibbiv/what-is-pub-sub-messaging-a-simple-explainer-2fdk).

Von Der Howen Georg. 2020. “Flutter Failed To Solve the Biggest Challenge for Our Cross-Platform App | by Georg von Der Howen | Better Programming | Feb, 2021 | Medium.” Retrieved February 17, 2021 (https://medium.com/better-programming/flutter-failed-to-solve-the-biggest-challenge-for-our-cross-platform-app-c551afa0ef18).

Jelvix. 2020. “(29) FLUTTER BY GOOGLE - PROS & CONS | WHAT IS DART? - YouTube.” Retrieved February 19, 2021 (https://www.youtube.com/watch?v=t5OCJQec0bk).

Lewis Jeffrey. 2019. “3 Ways to Secure Your Web API for Different Situations | by Jeffrey Lewis | The Startup | Medium.” Retrieved January 26, 2021 (https://medium.com/swlh/3-ways-to-secure-your-web-api-for-different-situations-8d5cd4762ab3).

Pluster Matt. 2019. “Realtime vs Near-Realtime Data: Pros and Cons.” Retrieved January 20, 2021 (https://www.skylinetechnologies.com/Blog/Skyline-Blog/November_2019/realtime-vs-near-realtime-data-pros-cons).

Ratros Y. 2019. “Authorization and Authentication in API Services | by Ratros Y. | Medium.” Retrieved March 8, 2021 (https://medium.com/@ratrosy/authorization-and-authentication-in-api-services-9b4db295a35b).

Singaram Muthu, and Prathistha jain. 2018. “Know the Difference Between Proof of Concept and Prototype.” Retrieved April 22, 2021 (https://www.entrepreneur.com/article/307454).

Wilson Christy. 2020. “The Difference Between Real-Time, Near Real-Time & Batch Processing.” Retrieved January 21, 2021 (https://www.precisely.com/blog/big-data/difference-between-real-time-near-real-time-batch-processing-big-data).

Annex 1: Material management plan

The diary is stored on drive C of the author's computer and is regularly backed up. The diary is kept at station C for at least one year after the completion of the thesis.