Fachhochschule
Nordwestschweiz

Swiss Engineering Event App (SEEA)

Projektarbeit IP5-FS19

Author:	Waleed Al-Hubaishi
Degree Program:	iCompetence
Coaches:	Prof. Dr. Doris Agotai
	Ms. Madlaina Kalunder
Client:	Mr. Thomas Hauser
	“Swiss Engineering Association”
Ort, Datum:	Brugg, 16. August 2019

Swiss Engineering Event App

Projektarbeit IP5-FS19

Autor

Waleed Al-Hubaishi

Pappelweg 7

8404 Winterthur

waleed.alhubaishi@students.fhnw.ch

Dozentin

Prof. Dr. Doris Agotai

Fachhochschule Nordwestschweiz

Doris.agotai@fhnw.ch

Ms. Madlaina Kalunder

Fachhochschule Nordwestschweiz

Madlaina.kalunder@fhnw.ch

Auftraggeberschaft

Swiss Engineering Verband STV

Thomas Hauser

Langackerweg 10

5303 Würenlingen

thomas.hauser@avaloq.com

Brugg, August 2019

1 Abstract

Swiss Engineering Association (SEA) has around 13’000 members with offices in Lausanne and Zurich. SEA offers its members the opportunity to develop their technical knowledge in the various engineering fields by hosting topic-centric events, which are attended by specialists in those topics to share their knowledge.

Those events aim to efficiently increase the knowledge level of the SEA members, and also to create smaller networks that consists of engineers from the same technical field or share similar interests.

Booking events can be done by visiting the SEA online platform, which can be accessed either via desktop or mobile browser. The website has a Mobile Responsive Design, however, finding the page to book an event is not intuitive, and the options available to filter results are limited.

The objective of this project is to facilitate the process of booking events using an app, so that the SEA members can totally rely on it to manage their booked events and explore what is available on the platform.

This project includes also an implementation of the Voice Assistant feature, to provide another way to interact with the app rather than the classical way of entering textual input via on-screen keyboard, although this implementation is limited for now to only one screen.

The app has been implemented using XAMARIN Cross Platform app developing tool to ensure that no matter which device do the members possess, they will still enjoy the intuitive experience of booking, managing and exploring the events.

Table of Contents

1. Introduction

Swiss Engineering is an association that spans over multiple professions and has a total of around 13'000 members in Switzerland. They are holding over 100 events annually over multiple of their Sections. These events are not only for current members also have the additional benefit of attracting new future members.

The Swiss Engineering association is currently in the process of updating their event booking website (see figure 1). To reach a greater audience they intend to also have a native mobile app to make booking on the go easier. This application would also be showcased on events with the idea to improve the image of the Swiss Engineering by showing that it has a modern app.

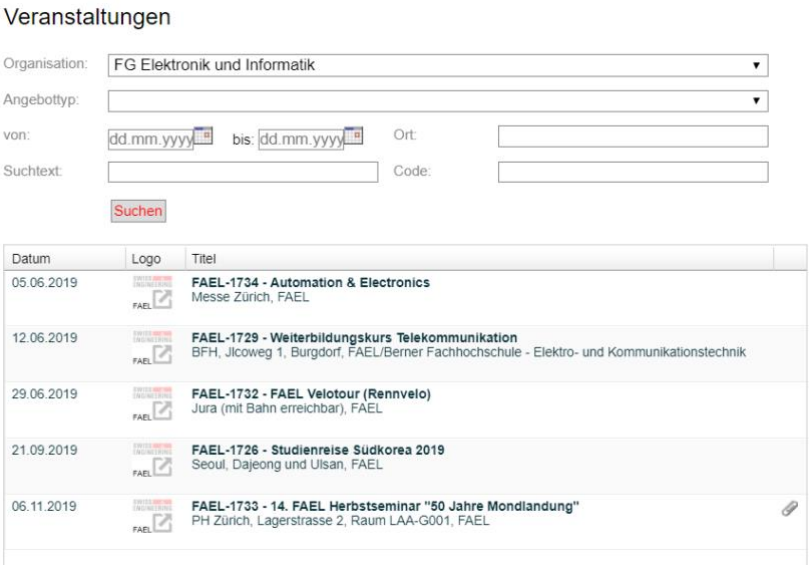


Figure 1 Swiss Engineering events platform

1.1 Problem Statement

The Swiss Engineering Website gives the users the possibility to search for events and book them online, but as the website is designed mainly to work on personal computer's browsers, the lack of mobile browsers support doesn't offer a great experience for either searching or booking events, as the user interface elements are not placed optimally on the screen and some are completely hidden unless the user scrolls.

In addition to this limitation, the website doesn't offer any kind of extra added values to the users, and the main and only function is to search and book events without any degree or possibility to customize or personalize the user experience while doing so.

The extra added value must help to retain the Swiss Engineering members, and make them tend to use the app in the future to book their events.

The Swiss Engineering Association also requested to have a voice assistant feature included in the app, with no specification on how such a feature shall be integrated or been used, so it is the duty of the team to find a user intuitive function, that can encapsulate the voice assistant feature.

1.2 Project Vision

The vision of this project is to build a simple User Interface Design to facilitate interacting with the app, with no need of any kind of guidance or tutorials.

User Interface elements, and app main features and functions shall be also clearly visible to the user. The user must not search for the features, but the features shall present themselves clearly to the user, which emphasizes the ease of use principle and aims to retain members.

The app shall provide an alternative to the textual input, namely speech input. The implementation of the user input via voice must be intuitive, therefor it must exist where the user is most likely to use it and not across the whole app's screens.

1.3 Project Approach

User-Centric design (UCD) is a process or set of tools used to design a service which focuses on what users need at the very beginning and continues throughout development until launch.[1]

One of the main goals set was to retain the members, the team decided to follow the User Centric Design (UCD) approach, as it focuses on the users and their needs in each phase of the design process. [2]

As specific requirements are not a part of the IP5, a sample of the Swiss Engineering potential members were interviewed to obtain their requirements and needs, which were then implemented in correlation with the business goals provided by the client.

2. Research

In the following sub-chapters the research work will be presented.

The research work covers the voice recognition methods, Speech-To-Text (STT) and Text-To-Speech (TTS) available Application Programming Interface (API).

The research work covers also the Cross Platform Integrated Development Environment (IDE) available.

Finally the available chatbots were also researched, in order to discover what options are suitable to be implemented in the coming versions to extract the user's information from the entered input, either via text or speech.

2.1 State Of the Art (Scientific Research)

This chapter discusses the state of the art models and algorithms that are used nowadays in the Speech Recognition field.

First, the most commonly used models will be explored, while illustrating their most remarkable features and techniques applied. Afterwards a more in depth look on how such models are being taught and trained to be able to identify the words and link it to the correct interpretation.

2.1.1 Voice Recognition Models

The following are the most commonly used models to recognize a speech. Those models have their own training techniques which converts the human input into a set of learning data, which are used afterwards to interprets the words.

- ***Automatic Speech Recognition***

The idea of Automatic Speech Recognition **(ASR)** is to translate the input voice into a written text. And because of the recent development, such algorithm exists nowadays on the majority of smart phones via the built in voice assistant apps such as Google assistant for Android devices and Siri on iOS devices, not only that but on every single smart speaker that contains the voice recognition such as amazon Alexa.

A neural network is a series of algorithms that endeavors to recognize underlying relationships in a set of data through a process that mimics the way the human brain operates. [3]

ASR models have benefited greatly from the development of deep neural networks, yet there is still a major challenge ahead. An example of those challenges is not being able to recognize a word, because the training data that contains this word has been dropped out.

Data augmentation, which is a way of creating new data from an original by manipulating its characteristics [4], has helped to overcome the previously mentioned challenges, the same way it helped before in the process of classifying images.

The way data augmentation is used in the process of ASR is to extend the number of data used to train, not by adding more learning data via adding new large sized files, but by manipulating the existing data to offer another version with different parameters (see figure 2), for instance the original voice can be used as one learning data, then speeding it up or lowering it down to provide two different new learning data. [5]

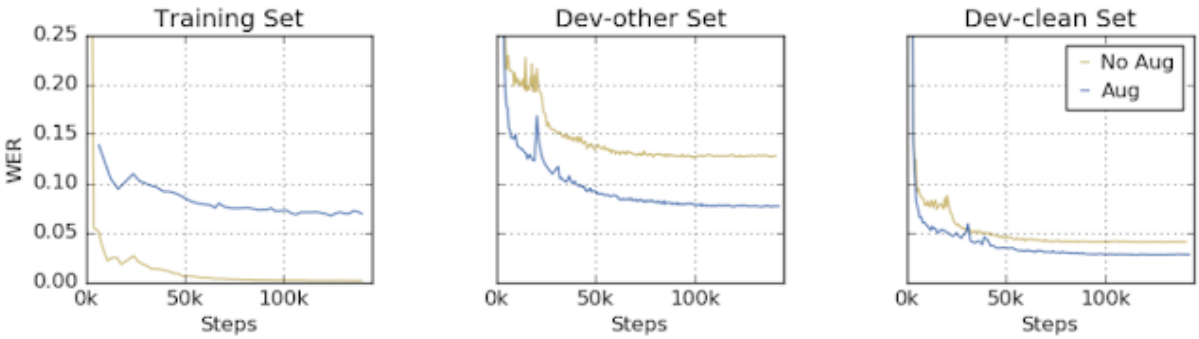


Figure 2 training data with two other versions created from original (https://ai.googleblog.com/2019/04/specaugment-new-data-augmentation.html)

- ***Multilingual Speech Recognition***

This model provides the possibility to decide, which language does a sentence belong to.

The algorithm idea is not new, rather it is an improvement over an already existed model (Listen-Attend-Spell attention based sequence-to-sequence ASR by William Chan) so that it can be adapted to help the purpose of differentiating the languages.

The idea of this algorithm is to include a set of languages, per say L1 … Ln in a set L, and also include their corresponding characters in a set for each C1 … Cn, also the language specific training data set which consists of (X1,Y1) for the first language till (Xn,Yn) for the language Ln.

A big set is constructed, that contains all the training data sets, and another set (C) which contains all the characters. Afterwards the big training data set is used to train the model, with no indication given which language does this training data set belongs to.

This test has been done over 9 Indian languages, which have a rare percentage of intersection of words and characters, so a small number of words may be used in more than one language, in this case the word X for instance will have most of the times only one language which it belongs to as the number of intersection is very limited, so the total number of words recognized will help to identify the language used according to the number of similarities of the recognized words and each language set (the set with the highest number of matches wins). [6]

At the end when the model finishes recognizing the words, a probability is calculated to determine to which language does this sentence belong.

This probability is calculated based on how many words came from each language separately, thus the language with the highest probability will most likely be the one which the sentence belongs to. [6]

- ***Visual-only recognition of normal, whispered and silent speech***

This is another state of the art model that might be helpful to use in the noisy or in contradiction the loud places is the visual speech recognition.

The idea behind the visual speech recognition is to not use the voice to recognize the words but rather to use the lip movement.

This model will not use the microphone or audio files to get its learning data but it will use the video instead to detect the lips movement (see figure 3) and link it to the desired word, but as the header indicates, this model is not only about silent speech, but also whispering and talking out loud as well as talking normal.

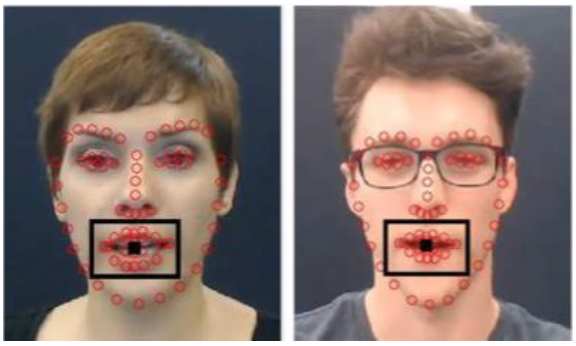


Figure 3 capturing the lips movement (Source: https://ibug.doc.ic.ac.uk/media/uploads/documents/normalwhispersilentdb.pdf)

The biggest challenge to implement this model was that the lip movement changes according to the context in which it is said, for instance the word “success” would have four different lips movement models, depending on whether it has been said silently, whispered, normal speech or said out loud. There is still a big chance that a word might be confused with another one in a different context if the mode is not set correctly while testing, or if the training data of a single context were not provided.

This is why it is required for this model, that the 4 different movements (normal, loud, whisper and silent speak) must be included in the training data set for the artificial intelligence (AI).

The results out of testing this model showed a big success and state of the art results when the context has been set and the training data has been provided while testing. On the other hand the performance dropped by a big margin when the model(context) has not been set, or when the training data of the used model was not provided but another model training data. [7]

2.1.2 Topic in depth

The voice recognition is considered to be another input alternative rather than the textual classical way of entering input to the machine via keyboard.

All different types of voice recognition requires two modes to work appropriately. The process starts with the training mode, followed by the testing mode.

Training mode

In training mode a huge amount of samples must be collected – the more the better – in order to train the system; no matter if it was speaker dependent or independent.

Speaker–dependent models operate by learning the unique, individual characteristics of a single person's voice, in a way similar to voice recognition.

Speaker–independent models are designed to recognize anyone's voice, so it requires no training. [8]

Acoustic/audible characteristics must be analyzed out of the input sample, and then the important features shall be extracted from it.

The feature vectors are then used to generate an input pattern which will be saved in a form of a matrix.

The samples must be words or even sentences captured by a microphone which is the input device in this case.

Testing mode

Similar to the training mode, features are extracted from the unknown entered pattern (voice input), then it is compared with all the input values in the matrix saved in the training mode, and the best match found should be considered to be the correct interpretation.

The following sub-chapters will discuss the various techniques of extracting features from the speech (i.e voice) input, then how those features are used to train a classifier so that it can classify the words spoken.

Feature extraction

In this section the different types of feature extraction techniques are covered, which will be used afterwards to train the system.

Feature extraction is done by changing the speech waveform to a form of parametric representation at a relatively lesser data rate for subsequent processing and analysis.

This process is also called “Front-End Signal Processing” and aims to transform the voice signal wave to a matrix that contains a numeric representation of the input voice’s features and characteristics (see figure 4). This makes the data more precise and refined. [9]

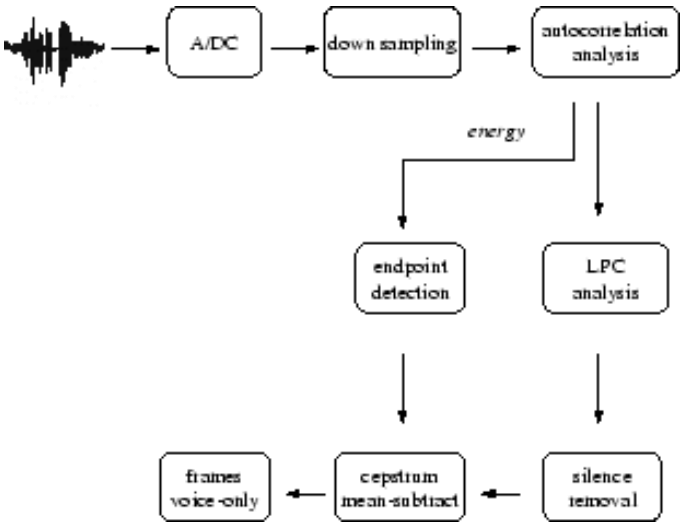


Figure 4 Outline of the front-end where LPC is the model used.
(source:https://www.usenix.org/legacy/publications/library/proceedings/sec02/full_papers/monrose/monrose_html/node3.html)

- ***Linear Predictive Coding (LPC)***

In LPC, the input speech in a frame-based manner is examined to produce vectors.

The LPC requires a pre-emphasize, which surpass the input speech, then the output of the pre-emphasizer must be blocked into frames, afterwards shall each frame be windowed, so that the amount of signal disruption can be reduced at the start and the end of each frame.

Finally each windowed frame is auto correlated, and the maximum value of the correlation is considered to be the order of the LPC analysis and used afterwards to return the LPC coefficient which is the result seeked (see figure 5). [10][11]

Figure 5 LPC flow diagram

- ***Perceptual Linear Prediction (PLP)***

It is similar to the LPC, but the captured speech must be adapted to the psychophysics of human hearing. Its spectral features differs from the normal LPC in way that matches the human auditory system, so therefor it rejects a lot inappropriate information (filtering) which might be caused by the speaker, but non-hearable by human frequency captured. [11]

- ***Mel Frequency Cepstral Coefficient (MFCC)***

The MFCC processing method is based on short-term analysis, which means that for each frame an MFCC vector must be computed, thus it is considered to be the best among all when it comes to estimate the human system response.

In order to obtain the coefficient (the result seeked in all types of speech recognition techniques) a hamming window must be applied to the input (speech sample) in order to reduce the disruption of a signal.

At the end a Discrete Fourier Transform (DFT) must be used to produce the Mel Filter Bank. [11]

MFCC doesn't offer only the possibility to recognize the speech, but also the speaker.

- ***Relative Spectral Filtering (RASTA)***

This technique serves the same purpose as the PLP, but it helps to drop out/filter the voices (frequencies) caused by the background noises that are not related to the speech itself, so if this technique was combined with the PLP, then the results should be brilliant, then the Rasta passes each feature coefficient only.

It also includes a linear filtering of trajectory of power spectrum in the case of the noisy speech. [12]

Words Classifiers

After extracting the voice features from the signals, those features should be used to train the system to classify the words spoken. Here is a list of the most commonly used classifiers.

- ***Hidden Markov Model(HMM)***

This model is trendy and an easy approach when it comes to classify words. It is based on huge vocabulary speech recognition systems, characterized by a finite state Markov model and a set of output distributions, and automatically trained on large speech data for many hours, thus comes its major advantage which is decreasing the time and the complexity required for training the huge vocabulary in the system. Unfortunately this advantage led also its major limitation, which is the complexity to find the error of its scheme in order to enhance the performance.[13]

- ***Neural Network(NN)***

Neural network are used mainly to solve complex identifications tasks, and they have also many number of advantages over the other previously mentioned models, such as its ability to function independently of the speaker (unknown speaker) and its ability to work with noisy data. When compared to the HMM, the neural network provides much more better accuracy, especially when the amount of training data is large as HMM is used mainly when the number of training data is limited.

The neural networks is being used also in phoneme recognition, and therefore a combination called NN-HHM does exist, where HHM is used for the language modeling, and the neural network for the phoneme identification part. [14]

- ***Dynamic Time Warping(DTW)***

Dynamic Time Warping uses dynamic programming to perform the optimization process of identifying the similarities between two samples, the first is the original, and the second is the manipulated version of it. That is why it has been used to identify the manipulated versions of voice, video or even images. [15]

- ***Vector Quantization(VQ)***

VQ is basically a function mapping process, where it maps a word or input from a large space, into a smaller space called cluster, a cluster is identified by a code word, those collection of code words construct a code book.

Using the VQ method, a new codebook is constructed for each speaker, thus it may recognize the speaker even. This constructed codebook (see figure 6) acts as a pre-recorded words for the user, then used when the speaker is being tested to be identified or to recognize what does the speaker say in the system.

When it comes to Voice Recognition, the VQ is used to retain the high speaker recognition rate as a parameter to identify elements like number of speakers for instance as well as the size of training database.[16][17]

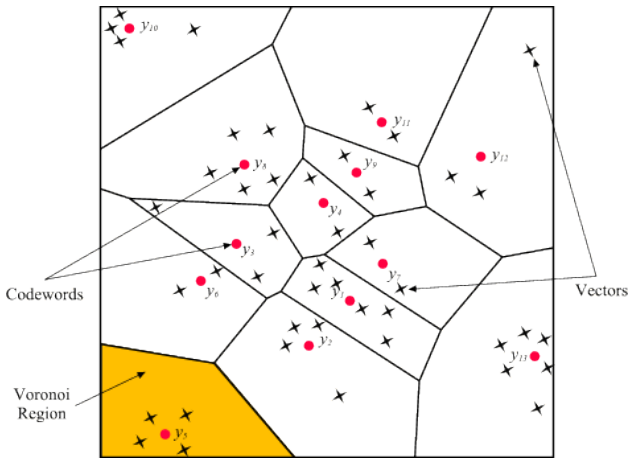


Figure 6 Codewords in 2-dimensional space. Input vectors are marked with an x, codewords are marked with red circles, and the Voronoi regions are separated with boundary lines (Source: http://www.mqasem.net/vectorquantization/vq.html)

2.2 Applications of Voice Recognition

This sub-chapter will go through the fields where the voice recognition is mostly used and in which context does it help to accomplish a specific purpose. At the first glance, one might think that voice recognition is used only for the smart assistant or voice messages, but it really did improve many other fields, that no one might consider the impact of voice recognition to those, and here are some.

- ***Evolving search engines***

The search engines nowadays have become a necessity in our lives, but people do find sometimes difficulties expressing what do they really mean in verbalized words, and would prefer to say our query more naturally, and that is the biggest role of voice recognition in the search engines.

It is not meant that queries shall be searched via voice, although that is also an option on the major search engine such as Google for instance, yet it is meant that the search engine use the voice recognition training data entered daily and the intent behind it to give back better search results with high percentages of accuracy. [18]

- ***Communication in service providers***

With the higher number of services available nowadays, therefor is the number of customers also huge, therefor the high number of technical support calls and guidance request, unfortunately that might lead to a long waiting periods on call till the customer been redirected to the technical assistant.

That has arisen the need of an innovative solution to help the customers get the guidance they needed in a short amount of time, thus is the voice recognition integrated.

In this situation the customer doesn't have to enter any numbers using the keypad, yet the customer should be able to speak more naturally and let the smart assistant guide him/her through the problem solving process. [19]

- ***Voice biometrics as authentication***

Voice biometrics captures multiple attributes of a person's voice, including the sound, pattern and rhythm in a form to construct what is also called a "Voice Print" that helps to identify the person. This model has been used even in Switzerland by some companies, Sunrise telecommunication for instance, as they would ask the customer to use a sample of the voice to get its biometrics and construct a model so that the customer doesn't have to give a number of personal sensitive data which might take a while before illustrating the customer's request and intent of calling.

The voice biometrics are not only used in the services company, but also to authenticate the unlocking process of the smartphone and also to give commands to the virtual smart assistant. [20]

- ***Smart Assistant***

Because of Amazon, Google and Siri, the smart voice assistant industry (see figure 7) is growing rapidly nowadays and companies compete to provide a better experiences to their clients to acquire a bigger market share.

Voice assistant helps to accomplish the major types of task, for instance setting up an alarm, playing music, calling contact, writing messages …etc.

Although a huge amount of people still consider the breakage of their own privacy, nevertheless the smart assistant is the considered to be the next big thing, especially when the companies start to expand the capabilities of the voice assistant as Google did for instant with the Duplex feature presented last year. [21]

Figure 7 available voice assistants on the market (source: https://idfive.com/ideas/preparing-website-voice-recognition-usability/)

- ***Forensic Department***

Voice recognition has helped a lot so far in identifying criminals using their voice, in this case if a sample of the criminals voice was captured as the crime was committed, this sample can use later on to identify the criminal among a list of suspects by comparing their voices to the captured sample. [22]

2.3 Technical research

This chapter will discuss the major technical issues related to the decision making processes during the project developing phase.

2.3.1 Voice Assistants

This section will go into depth about which existing implementations for Voice Assistant applications are available, and their major features and characteristics

- ***Google assistant***

Google assistant is a platform independent natural language processor. It supports both Android and iOS devices.

It is recommended to use DialogFlow (previously api.ai) for the natural language processing in the background. DialogFlow (see figure 8) tries to extract intents and entities from the input.[23]

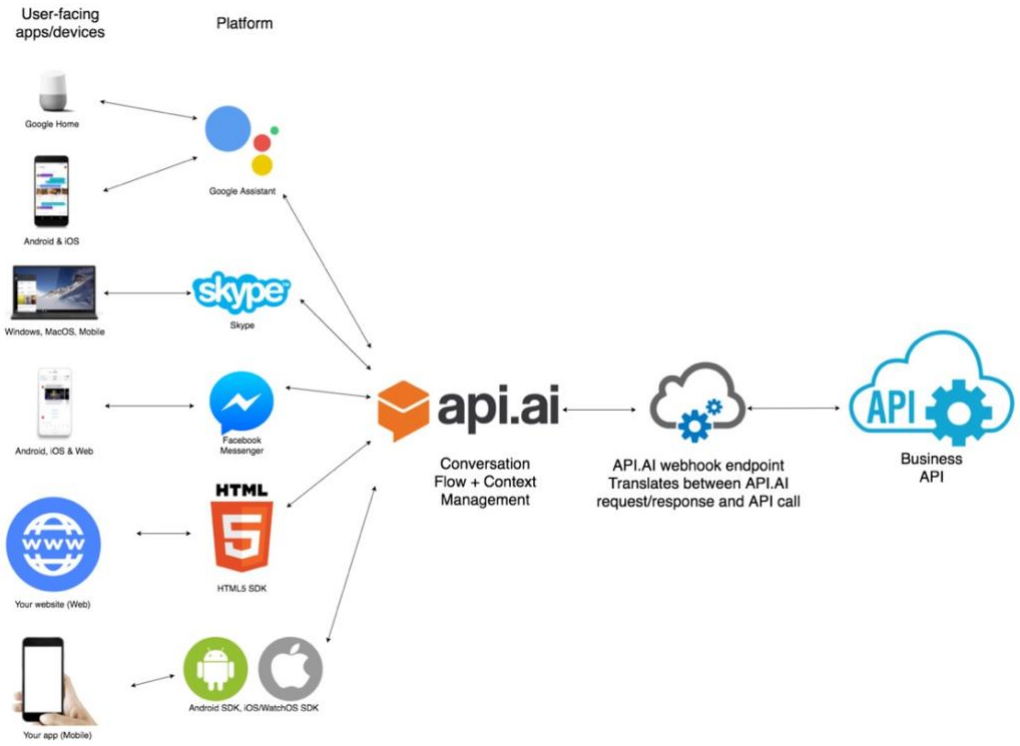


Figure 8 DialogFlow(api.ai) structure (source: https://labs.kunstmaan.be/blog/hey-google-what-can-you-do)

Entities could be for example Entities are Dialogflow's mechanism for identifying and extracting useful data from natural language inputs. [24]

The Entity Animal contains multiple different animals, e.g. dog, cat, bird, etc.

After creating a list of things that are part of the “Animal” Entity you can also help the A.I. by providing synonyms. E.g. A “Puppy” is also a Dog, “Dogs” can also be interpreted as Dog, etc.

Intents are actions that can be derived from phrases the user tells the program. E.g. “Tell Me A Joke” would be an intent, the program will be prompted to execute the task (intent). Like for entities, the developer needs to provide the algorithm with example sentences that should trigger the intent. The more examples are given, the more non predefined sentences the A.I. can use to trigger the intent.[25]

Domains[26]

- Control devices and smart home.
- Access information from calendars and other personal information.
- Find information online, from restaurant bookings to directions, weather and news.
- Control music.
- Play content on Chromecast or other compatible devices.
- Run timers and reminders.
- Make appointments and send messages.
- Open apps on phone.
- Read notifications.
- Real-time spoken translations.

Pricing

Dialogflow Standard Edition provides all of the core features of Dialogflow, but interactions are limited by usage quotas, and support is provided by the community and e-mail. It is ideal for small to medium businesses that want to build conversational interfaces or those who want to experiment with Dialogflow.

Dialogflow Enterprise Edition provides higher usage quotas and support from Google Cloud support. Dialogflow Enterprise Edition is a premium offering, available as a pay-as-you-go service. It is ideal for businesses that need an enterprise-grade service that can easily scale to support changes in user demand.

Available in two pricing plans:

Essentials This plan contains all features offered by Dialogflow Standard Edition, plus enterprise-ready quotas for speech recognition, speech synthesis, and telephony gateway.

Plus This plan contains all features offered by Essentials, plus enterprise-ready quotas for knowledge connectors. Each request from an Enterprise Plus agent performs the regular

intent recognition and entity extraction, as well as a knowledge connector search. See figure 9 for more details.

	Free Standard Edition	Pay as you go Enterprise Edition	
		Essentials	Plus
Knowledge Connectors (Beta)	Limited	Limited	Unlimited*
Text or Google Assistant	Unlimited*	Unlimited* $0.002 per request	Unlimited* $0.004 per request
Audio Includes speech recognition and synthesis (Beta)	Limited	Unlimited* $0.0065 per 15 sec of audio	Unlimited* $0.0085 per 15 sec of audio
Phone Call (Beta) Includes phone connectivity, speech recognition, natural language understanding, speech synthesis	Limited	Unlimited* $0.05 per min of phone call processed	Unlimited* $0.065 per min of phone call processed
Toll-free phone call (Beta)	None	$0.06 per min of phone call processed	$0.075 per min of phone call processed

Figure 9 DialogFlow Pricing (source: https://dialogflow.com/pricing)

- ***Siri Assistant***

Siri is Apple's voice-controlled personal assistant and it has been around for several years now.

The development environment of Siri is called SiriKit (see figure 10) and it offers the possibility to translate calls to the assistant into calls for the app you want to use. For this to work an internet connection is mandatory as the voice data is sent to apple servers for the natural language processing.

Apple has a strict List of so-called Domains and Intents which the application call has to be assigned to for it to work.[27]

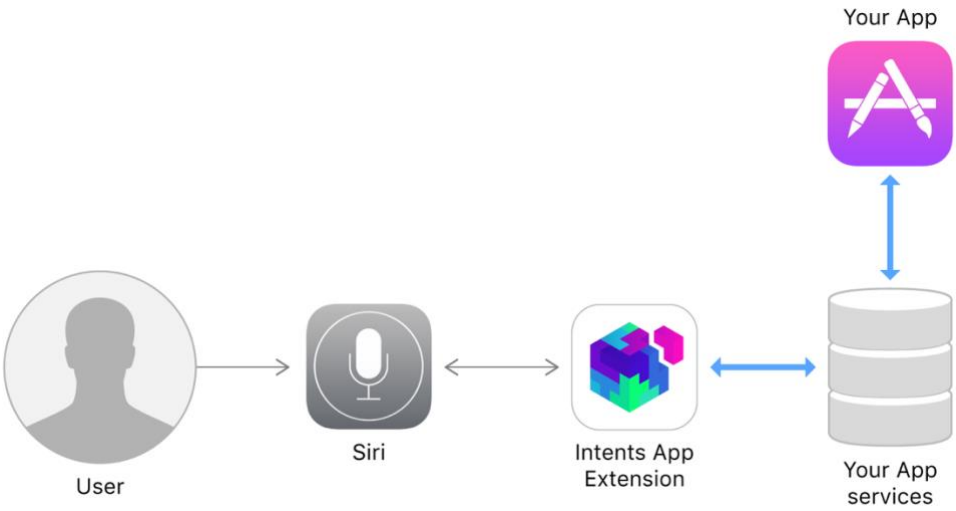


Figure 10 Making services avaliable from Siri and Maps (source: https://developer.apple.com/documentation/sirikit)

Domains[27]

- **Lists**: Simple list creation and editing
- **Visual Codes**: Display QR codes
- **Ride Booking**: Requesting a ride, this is intended for services like Uber
- **Messaging**: Simple text messaging
- **Photo Search**: Making a search request for photos and videos and display them.
- **Payments**: Creating a payment request or sending money
- **VoIP Calling**: Make a call (also supports video)
- **Workouts**: Starting, pausing and finishing a routine, explicitly specified as workout
- Climate and radio.

Pricing

The pricing of Siri API is dependent on the Tier of choice. See figure 11 for details.

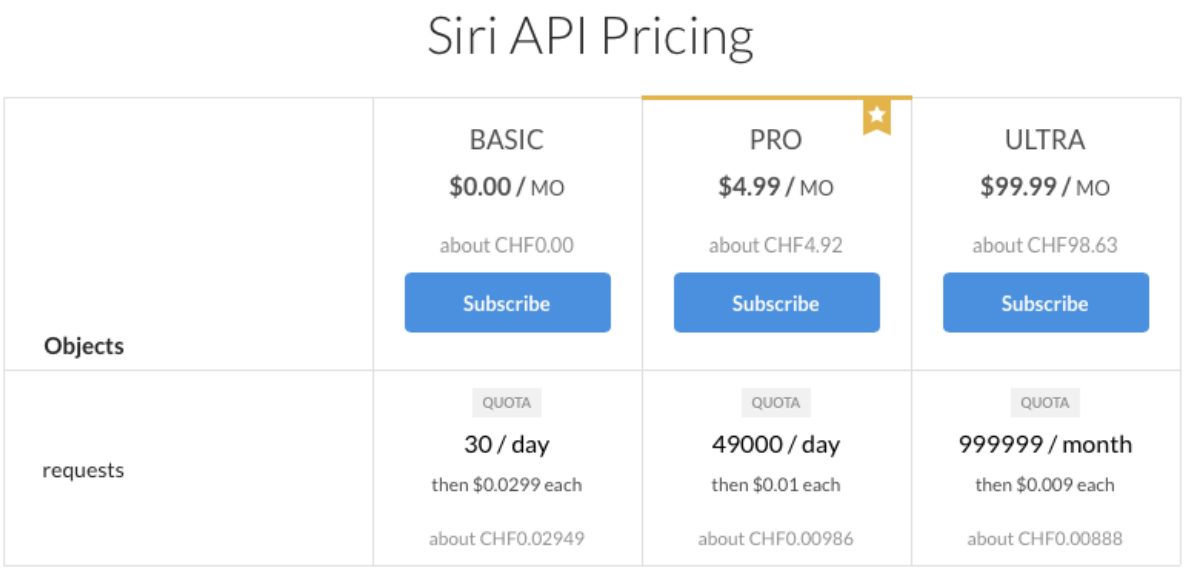


Figure 11 Siri Pricing (source: https://rapidapi.com/pannous/api/siri/pricing)

- ***Alexa Voice Service (AVS)***

AVS (see figure 12) is a platform independent service that is used over an HTTP/2 api.[28]

AVS is task oriented. A client interacting with the Alexa Voice Service will regularly encounter events/directives that produce competing audio. [29]

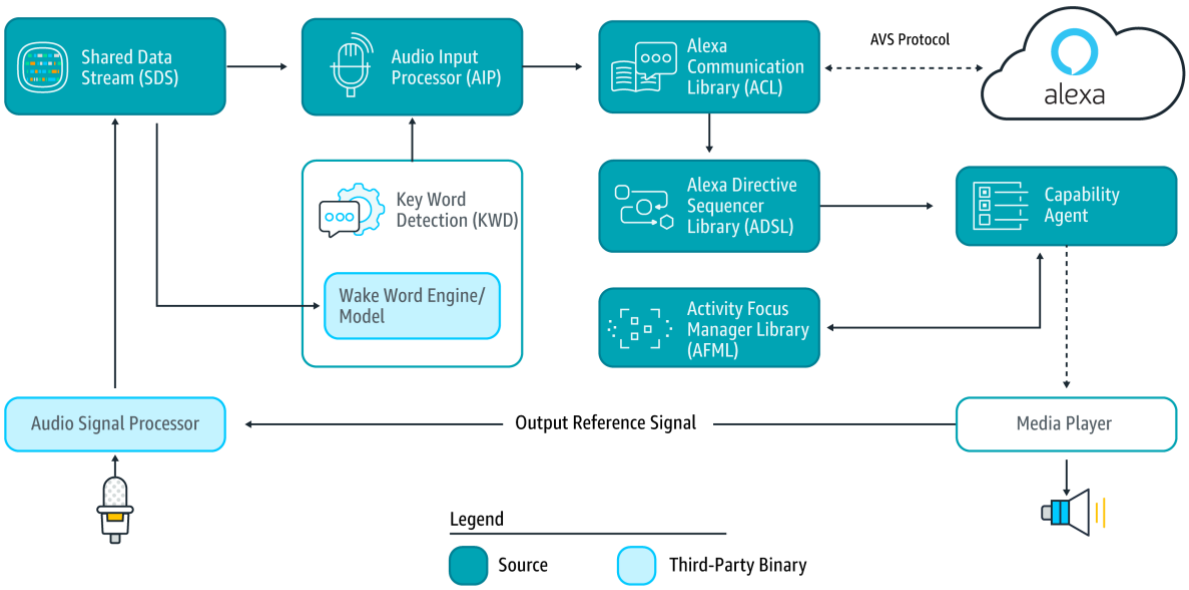


Figure 12 AVS structure (source: https://www.digikey.in/en/articles/techzone/2018/feb/rapid-prototyping-smart-voice-assistant-raspberry-pi)

Capabilities[30]

- **Alerts**: timer and stopwatch features.
- **AudioPlayer**: control music playback, etc.
- **Bluetooth**: manage Bluetooth connections.
- **Do Not Disturb**: enable Do Not Disturb mode on device.
- **Equalizer Controller**: control equalizer settings and equalizer modes.
- **Interaction Model**: Allow the client to support complex interactions and Alexa routines.
- **Notifications**: API for notifications.
- **Playback Controller**: navigate playback queue via GUI or buttons.
- **Speaker**: volume control, mute and unmute.
- **Speech Recognizer**: API for speech capture.
- **Speech Synthesizer**: Text to Speech API.
- **System**: System state.
- **Template Runtime**: visualize metadata of requests.

Pricing

The pricing of AVS is dependent on the Tier of choice to apply. See figure 13 for more details.

Input requests to process	Cost per request	Number of requests	Total
4,000 speech requests	$0.004	4,000 requests	$16.00
1,000 text requests	$0.00075	1,000 requests	$0.75
Total Amazon Lex charges for 4,000 speech and 1,000 text requests			**$16.75**

Figure 13 Amazon Voice Services (AVS) pricing (source: https://aws.amazon.com/lex/pricing/)

2.3.2 Plugins for Text-To-Speech

Beside the built-in Text-To-Speech APIs, there are also multiple plugins that could be used to generate voice using text, as also called Text-To-Speech (TTS) plugins.

- ***Xam.Plugins.TextToSpeech***

This plugin has been developed a famous Nuget plugins developer “James Montemagno”, and it is a simple and elegant way of performing Text To Speech across Xamarin.iOS, Xamarin.macOS, Xamarin.Android, Windows and Xamarin.Forms projects. [31]

This plugin has been downloaded about 1.5M [31] times till the time of writing this report.

Platform Support

The supported platforms are listed in figure 14.

Platform	Version
Xamarin.iOS	iOS 7+
Xamarin.Android	API 10+
Windows 10 UWP	10+
Xamarin.Mac	All
Xamarin tvOS	All

Figure 14 supported platforms (source:https://github.com/jamesmontemagno/TextToSpeechPlugin)

- ***Google.Cloud.TextToSpeech.V1***

Is recommended Google client library to access the Google Cloud Text-to-Speech API, synthesizes natural-sounding speech by applying powerful neural network models. [32]

This plugin has been downloaded 16.5K [32] times till the time this sub-chapter has been written.

The usage of this API is not straight forward as it requires some extra pre-usage steps and authentications. [28]

Platform Support

Google claims that they are trying to support as much set of environments as possible [29], although they have acknowledged that there exists some issues regarding Universal Windows Platform (UWP). [33]

- ***Snow.Xam.Plugins.TextToSpeech***

This plugin has been developed by “Ivan Petrov” and it is described by the developed as a plugin for Xamarin and Windows to perform text to speech functionality. [34]

The plugin has been downloaded 893 times [34] till the time this sub-chapter has been written.

Supported platforms

Not specified by developer. But it is worth mentioning that this plugin has not been updated since august 2017. [34]

2.3.3 Cross-platform Frameworks

This chapter will discuss the major available options to choose from when it comes to develop a Cross-Platform mobile application developing framework.

- ***XAMARIN Forms***

Xamarin is a framework and a set of tools to make it possible to do native Android, iOS & Windows development in C# (.Net).

With a **C#-shared** code base, developers can use Xamarin tools to write native Android, iOS, and Windows apps with native user interfaces and share code across multiple platforms. Xamarin integrates with Visual Studio, Microsoft's IDE for the .NET Framework, extending Visual Studio for Android and iOS development. [35]

Features

- Native UI, native API access and native performance.
- Anything you can do in Objective-C, Swift, or Java you can do in C# with Xamarin.
- Cutting-edge apps with same-day support for new OS releases. [35]

Cons

Using Xamarin, you still won't be capable of using numerous open-source libraries for iOS and Android development because of **compatibility issues**. The **free version is seriously limited** for developing a substantial project. [35]

- ***PhoneGap***

PhoneGap is best suited for mobile applications that **don't make substantial use of the phone's native features**. It packages your applications within **a native application container** which allows **JavaScript** to access device-level APIs the same way normal apps do. [35]

Features

- It allows creating **hybrid apps using** popular web technologies (HTML5, CSS3 and JavaScript), which are also readily available skill sets
- It lets you deploy a single code base to different platforms including iOS, Android, Windows Phone, BlackBerry, Firefox OS and more
- It follows a plugin-able architecture, which means that access to native device APIs and more can **be extended in a modular** way. [35]

Cons

The performance is known to be a bit poor for graphic-intensive apps made using PhoneGap. Though you may easily find plugins for PhoneGap per requirements, they could be somewhat outdated or unsupported based on the target platform(s). [35]

- ***Codename One***

Codename One is a cross-device platform with goals of **simple usability, rapid application development**, deep integration with the native platform with possible native speeds. While you're required to **code in Java**, your application can also be tested & verified with Codename One's simulator devices and test automation tools. [35]

Features

- It *supports most of the popular IDEs* ***like NetBeans, Eclipse, IntelliJ IDEA, etc.***
- Its 'lightweight architecture' allows the UI to work seamlessly across all platforms
- Its build servers allow building native iOS apps without a Mac machine and native Windows apps without a Windows PC
- It uses ParparVM which guarantees compatibility with future iOS versions because of its use of the officially supported iOS toolchain. [35]

Cons

Codename One's default **visual themes are a bit primitive** and its **Graphical UI Builder may not be very suitable for large projects**. With growing project's complexity, maintaining a single file with all event handlers becomes very cumbersome. [35]

- ***Gluon Mobile***

A new open source deep learning interface which allows developers to more easily and quickly build machine learning models, without compromising performance. Gluon provides a clear, concise API for defining machine learning models using a collection of pre-built, optimized neural network components.

Gluon is a tool in the **Machine Learning Tools** category of a tech stack.[36]

Features

- Gluon offers a full set of plug-and-play neural network building blocks, including predefined layers, optimizers, and initializers.
- Gluon enables developers to define neural network models that are dynamic, meaning they can be built on the fly, with any structure, and using any of Python's native control flow.
- Gluon does not require the neural network model to be rigidly defined, but rather brings the training algorithm and model closer together to provide flexibility in the development process.
- Gluon provides all of the above benefits without impacting the training speed that the underlying engine provides. [36]

Cons

Lack of developers community engagement, documentation and online tutorials. [36]

- ***Flutter***

Flutter is a free, open-source mobile SDK that can be used to create native-looking Android and iOS apps from the same code base. Being in beta for a while, Flutter 1.0 was officially launched in December 2018. [37]

Flutter apps are built using Dart, an object-oriented programming language.

Features

- Fast performance as it is using what is called “Hot-reload” which injects the updated source code file(s) into the running Dart virtual machine.
- Expressive and flexible user interface as it includes a full set of widgets that deliver pixel-perfect experiences on both iOS and Android. And it enables the ultimate realization of Material Design, Google's open design system for digital experiences.
- Widgets support provides view and interface to an app provides a natural look and feel regardless of screen size. Widgets provide features like navigation, scrolling, icons, and fonts to provide full native performance on both iOS and Android.[38]

Cons

- Flutter caters to mobile apps only and isn’t supported by web browsers.
- Limited libraries as it is still relatively new.
- No support for either Apple or Android TV.
- There are ready-made Flutter solutions for most popular CI platforms like Travis or Circle. However, the toolkit must be set first for these kinds of platforms. [39]

- ***Progressive Web App***

A Progressive Web App (PWA) is a platform developed as a web application that acts as both a website and an application.

PWA's apps appear as a website while browsing, but act as applications as well with an application's icon on the mobile's and tablet's homepage.[40]

Features

- PWAs are responsive to any and every device, whether it was a desktop, tablets or a mobile.
- They work for all web-browsers, which means they have the reach of websites while having the capabilities of native apps.
- Load an entire page/application in offline.
- Secured via HTTPS coding.
- Uses the same platform that the actual website was built on. [40]

Cons

- PWA can not access the native applications on the device (e.g calendar ,camera …etc).
- No Cross app login, which means login to a PWA using Facebook or Instagram accounts is not yet supported.
- The apps are not available on the native applications store. [40]

2.3.4 Chatbot service platforms

This following table explores the various chatbots platforms available, which may host SEEA interaction questions, and the Tags extraction process. It indicates their major differences.

Chatbot	Ease of use	Integration	Cost	Supported languages
Dialogflow previous known as **API.ai** is Google-owned. [41]	Uses web interface to create bots. Basic attributes such as Intents, Entities and Actions are easy to configure. [41]	• Google Assistant. • Websites. • Slack. • Facebook Messenger. • Skype. … Etc. [41]	Standard edition is Free. Enterprise plan starts with 0.002$ per request. [41]	20 languages. [42]
Amazon Lex is powered by the same deep learning technologies as Alexa. [41]	Uses web interface to create and launch bots. [41]	• Facebook Messenger. • Kik. • Slack. • Twilio SMS. [41]	Free plans are available for the first year with limitation. Voice request is charged at $0.004/request while text requests are priced at $0.00075/request. [41]	US English.[43]
IBM Watson Assistant supports searching for an answer from the knowledge base. [41]	Easy to navigate user interface displayed on a web browser. [41]	• Facebook Messenger. • Telephony. • Slack. • WordPress. • Custom apps using API.[41]	Free plans available. Paid plan starts with Standard at $0.0025/message.[44]	All supported languages are in Beta. 10+ languages.[45]
Wit.ai used for creating intelligent bots. [41]	Easy to navigate user interface displayed on a web browser. [41]	All integrations through HTTP API and libraries in Node js, Python, Ruby and Go. [41]	Wit is completely free of charge. [46]	50+ languages. [41]
Azure Bot Service Owned by Microsoft and used for creating intelligent bots. [41]	Uses web interface to create and launch bots. [41]	• Facebook Messenger. • Skype. • Slack. • Microsoft Teams. [41]	Free plan available. $0.5/1000 message. Extra charges for consumed resources. [41]	5+ languages. [41]

3. Concept

This chapter will discuss the approach perused while designing the app and important features introduced in the solution.

3.1 Approach

While designing the app a mixture of Brainstorming, existing platform research and User-Centric Design was used.

The Swiss Engineering platform (the version that was in use before mid-May 2019) was researched to construct the requirements for most of the features implemented.

The Brainstorming design approach was used mainly to construct the requirements for the voice assistant feature requested by the client, as well as small addons to provide extra value for the end-user while interacting with the app. More details follows in the Requirements sub-chapter.

The User Centric Design (UCD) was used to figure the design structure and the information flow that would make the most sense for the end-users while using the app.

A sample of potential end-users was heavily engaged while designing the app user interface, and their preferences were taken seriously into account even while choosing the color palette of the app. The full story of app design can be found in the appendix chapter.

The UCD approach power is its tools used to develop an understanding of the users to a specific level, where the team can then carry on with the design process using the brainstorming method to come up with a list of features that will help to satisfy the user's needs, by evaluating the solution against the requirements.

3.2 App Map

The screen navigation of the Swiss Engineering Event App (SEEA) is described in figure 15, where each box is a screen, completed arrow is a navigation line and the dotted/dashed lines are either contained pages or conditions that must be fulfilled in order to be able to navigate.

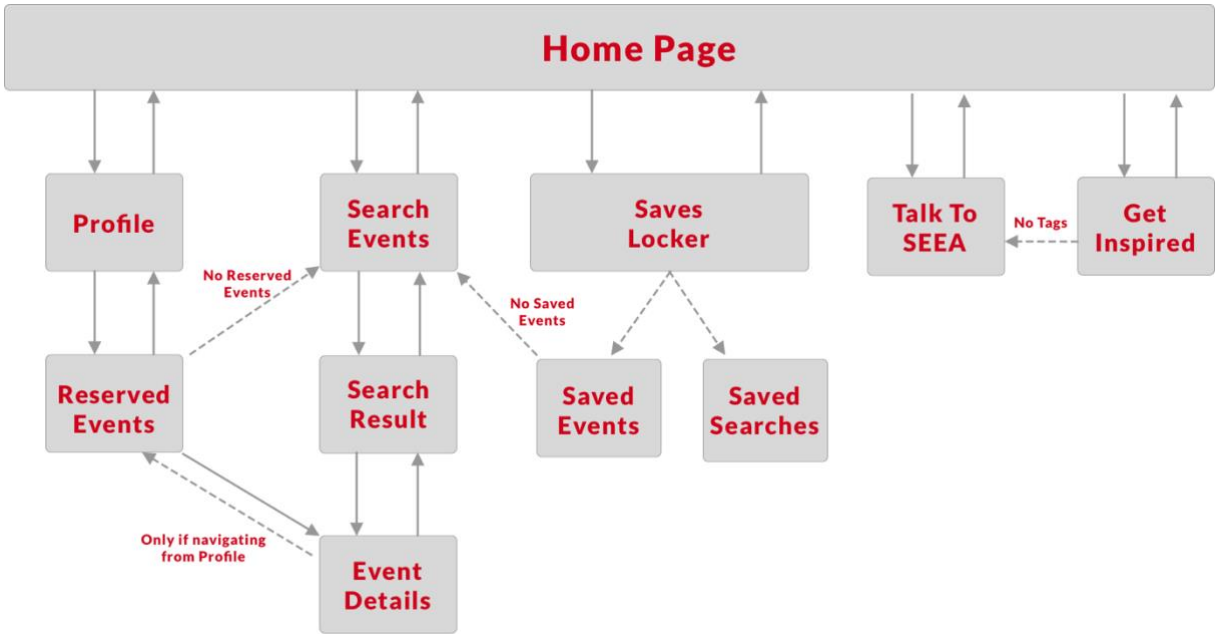


Figure 15 Swiss Engineering Event App map and screen navigation

3.3 Persona

The team did not have access to the Swiss Engineering members data for privacy reasons.

The team had therefor to interview the sample of the potential end-users to gather information about their preferences and characteristics in order to construct a persona. The client was also asked to give input regarding the members of Swiss Engineering; age and proficiency for instance.

Hypotheses were also made to cover some aspects of the persona's personality that were not covered by the interviews.

According to the client's input, a hypothesis proposed that the app is targeting the Swiss Engineering members, who are mainly engineers starting mid 30's, looking for events which might give them a quick brief into what is new and available in the market, so that the members can choose then to go deep into the presented topic in the events or pass it and look for another topic that may be interesting.

Another persona were constructed to address the non-member end-user, assuming that the app will still be available to download for all people who have access to either Android or iOS app store.

For further details, the Member and Non-member personas can be found in the appendix chapter.

3.4 User Interface

All the main features that the app presents are located on the home page, and the user will be asked to return to the home page whenever a navigation from a feature to another is needed.

The app features are ordered based on their degree of importance to the app on the home screen (up to bottom).

The features are presented this way so that the end-user does not struggle searching for the function requested, and to have a better overview about the capabilities of the app from the start.

A shade of the red color is used as the main brand color, where a shade of “cyan”, which happens to be the inverse of that red shade, is used to highlight the clickable buttons that does not have a great impact on the user experience.

3.5 Requirements

The requirements were gathered using both the platform (used before mid-May 2019) research and end-user’s (sample of potential end-user) as well as client’s interviews.

The platform was researched to figure the set of features that the platform offers to its members, so that the main features will still be included in the application. The reason for that is to retain the familiarity, so that the end-user who has used the platform before would still be able to navigate through the app screens and could accomplish their goals.

The Swiss Engineering platform was also researched in order to gather the information required by the Swiss Engineering server to retrieve results for some operations (e.g Search for event query).

Brainstorming design approach was used to evolve a usage for the voice feature requested by the client, as well as small addons that shall provide extra value for the end-user while interacting with the app.

The new ideas that were brainstormed by the team members were evaluated according to their feasibility. The criteria of this evaluation process were time, man-power resource and the client’s infrastructure (e.g Database design).

The sample of end-user was also personally interviewed to evaluate the brainstormed ideas against their requirements, and whether or not would they use those features.

Verdict and Impressions

- All the end-users interviewed expressed that they will never use the voice assistant feature for instance to search for events, rather search via text input. The reason is that they usually use those applications while they are in their offices or public areas, and talking to their assistant would attract unintended attention. For others they mentioned that speaking out loud would violate their privacy and expose personal information.
- The short scope of voice assistant, as it is only responsible of extracting the tags from the sentences, led to skip using a chatbot for this version as a voice assistant controller and responses generator, and rather extracting the tags via Contains() and After() methods then saving them into the users own profiles.
- The end-users interviewed also mentioned that they would rather have all the search criteria entered in one screen rather than splitted into multiple screens, where each has its own criteria to be entered, in order to save time and not having to tap multiple times to reach their results, therefor an old design was altered in order to address those inputs.
- Facebook Marketplace, YouTube and even Ricardo do all have a section where content is being suggested to the user, that might be found interested. So such a feature is important to have, as it would help give the user an exceptional personalized experience as it recommends events using the personal liked tags/topics, and also help the client to get more events booked.

3.6 Social Factor

According to resent Harvard Business Review, engineers had the second loneliest profession, where lawyers take the lead.

The findings may not come as a big surprise. Engineers spend a lot of time working on their designs in front of their computers. Technology allows them to collaborate and coordinate with colleagues instead of meeting with them face-to-face. Their work environment doesn’t always allow them to form friendships with colleagues.

The Swiss Engineering event app has also this aspect considered, and tries to provide a an anti- loneliness platform to the Swiss Engineering Association members.

4. Prototype

The main feature that the application is built on called “Tags System”.

- ***Tags System***

The Tags System idea is to create a set of tags corresponding to each events, those tags help to identify the content of each event and to which technical field it belongs.

The tags can be constructed from the description of each event, or entered manually as each event object is created by the Swiss Engineering platforms admin.

Those tags can be used later on to refer events to members interested in those tags, or to filter the results of event searching process.

The Tagging System is the core of implementing the voice assistant feature in the Swiss Engineering event app, as there is no other way to save tags into the users profile but with interacting with basic AI called SEEA.

The importance of the Tagging system is that it offers the ability to classify the events into more useful and meaningful clusters, and will help to retrieve even better results when it comes to search for events, as well as suggesting relevant events tailored to the users events and past booking. The extra system business value is that it offers the possibility to know which types of events are the end-user mostly interested in, so that the client might have an insight and can optimize the attendance rate and minimize the no-shows.

The following sub-chapters discuss the set of features presented by the Swiss Engineering Event App including Core and Explorative features in details, as well as the design characteristics and schematic UX design.

4.1 App main features

The set of functions to be presented in the app are as follows.

- ***Talk To SEEA***

Where the end-user will be able to interact with the built in voice Assistant to enter the preferred Tags, so that the suggestion the Get Suggestion Page improves.

- ***Get Inspired***

Where a list of suggested events based on the tags entered to SEEA been shown. And also the events that has been suggested based on the Social Factor that has been discussed previously.

- ***Search events***

Search event is the core feature, no event app can be accomplished and no good level of user experience can be offered without including this functionality.

- ***Profile***

The profile page will include all the personal data of the user, beside a tab to navigate to the booked events and a Dark theme toggle button to enable the Dark mode, to address the sample of end-users who liked this theme.

- ***Saves Locker***

Saves Locker is the feature that will enable the end-user the possibility to save events and searches, so that it is all kept under one place, but elegantly separated to enable the ease of access to the required data fast.

In the following sub-chapters, the core features implemented as well as some explorative features are discussed.

4.2 Core implemented features

Both “Tags system” and “Talk To SEEA” are related, as the voice assistant feature included extracts the tags from the user input, either via voice or text, and then tries to suggest new events to the end-user by comparing those extracted tags and the tags included in each event.

- ***Talk To SEEA***

Talk To SEEA has two modes, the first is the text mode, in which the end-user will choose to enter the input textually, and SEEA shall respond also in text. The end-user might choose also to give the input via voice, and in that case SEEA will also respond via voice, but both end-users and SEEAs input will still be shown as text, so that the end-user might still check if SEEA got what was said correctly, and that the end user may refer to the textual output in case the end-user didn’t get what SEEA said correctly.

Talk To SEEA enables the end-user to have a conversation with the built-in Assistant (see figure 16 and figure 17), so that the end-user may enter the Tags that the end-user find interesting, to use those tahs then to re-direct events to the Get Inspired page, in order for them to be suggested to the user.

Figure 17 Talk to SEEA via text

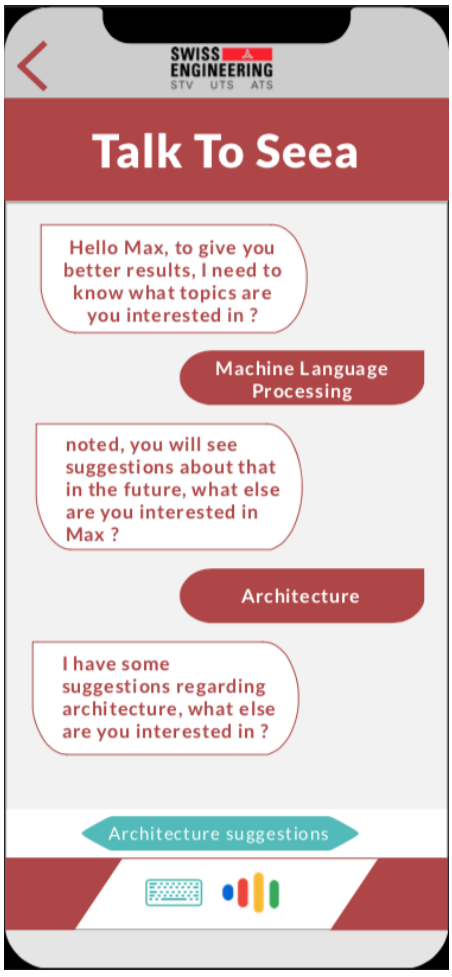


Figure 16 Talk to SEEA via voice

If the end-user entered Tags were found, SEEA will show also a small action button in the suggestions bar that opens the Get Inspired page.

In order to implement the Voice Assistant, the Xam.plugins.TextToSpeech is used to generate the voice out of the textual result generated by the simple AI implemented, due to its good reputation (Number Of Downloads), simplicity and no pre-usage conditions nor authentication like Google.Cloud.TextToSpeech.

- ***Create familiarity in society***

The idea of the Co-Attendees is to recommend events to an end-user based on a Social Factor.

This idea was formalized to take a step, and act against the loneliness spread among the Engineers Society. It will help the Swiss Engineering members to see familiar faces in each events, so that the member can then approach those people and get to know them and make friendships inside the Swiss Engineering Society, under the slogan “From Virtual to Reality”.

If a member called “Lorenzo” for example attended an event with another member called “Sarah”, then the future bookings of “Sarah” will be suggested to “Lorenzo” to attend (see figure 18). This all happens anonymously, so “Lorenzo” will never know based on whom was this event suggested to him, rather he will find those events labeled and listed under the Social Factor findings.

Of course the members are not forced to get to know those familiar people, but it is rather just a push from the Swiss Engineering towards this step, while the final decision is totally up to the end-user.

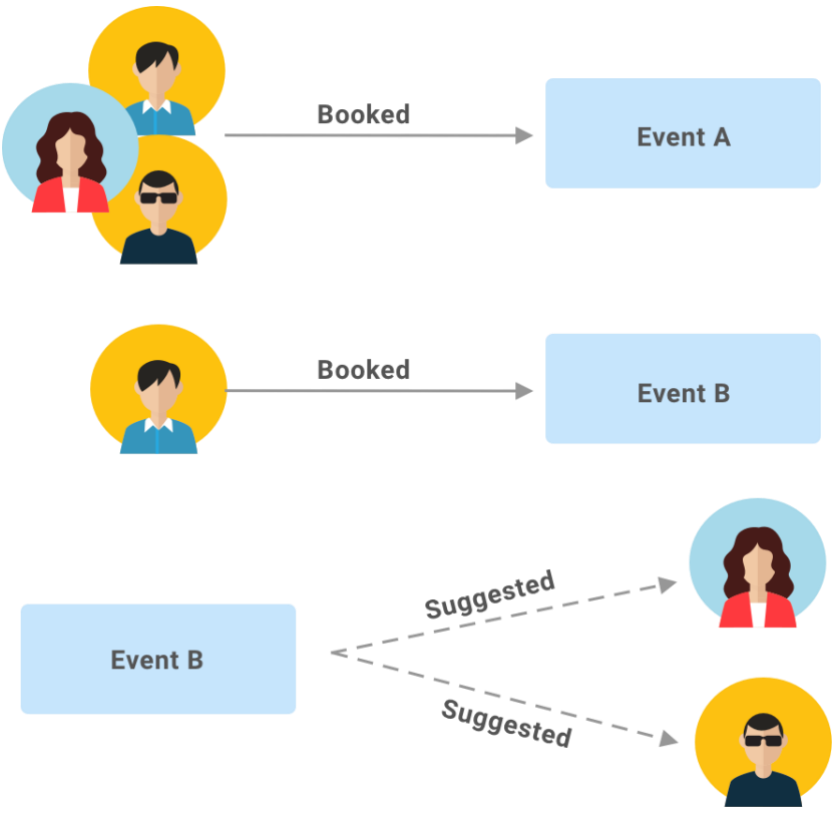


Figure 18 suggest based on ex co-attendees

4.3 Explorative features and functionality

This sub-chapter discuss some the features and functions that were explored during the early phases of the app design and prototypes, but were not feasible to implement in the first version of the app, rather in future versions. For the full list of features explored refer to the appendix chapter.

- ***EventTinder***

The Idea of EventTinder is to swipe right for the tags that the end-user is interested in, or to swipe left in case the end-user has chosen to not hear anything about this tags(topic) anymore (see figure 19).

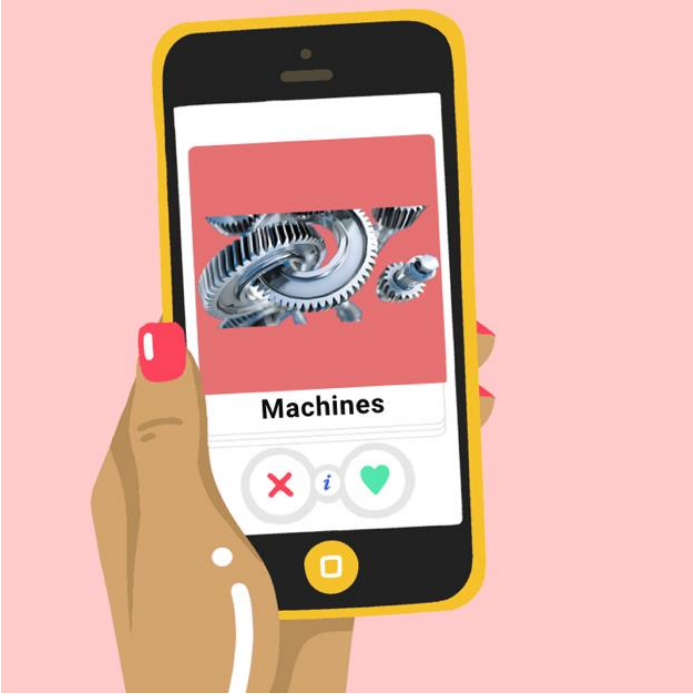


Figure 19 EventTinder

- ***TagsSquare***

The idea of TagsSquare is to drag and drop the visible card into one of the four corners, where each corner represent the end-users level of interest about the name shown on the card (see figure 20). The events that belong to the tag will be often suggested based on the side selected, where by green suggest every single event that contains this tag, and never by red.

Figure 20 TagsSquare

- ***Search Event by voice***

Using this feature, the end-user will enter the search criteria via voice, so the user may say “Search for user design events in Zurich” and the app must retrieve the results and show it on the screen if any. But it was removed afterwards as the intuition of this feature was questioned, and after the feedback of the end-user which stated that they will most probably not use it, and rather search for events using the classical textual input approach.

- ***Add to my Calendar***

The idea is to simply ask the enabled voice assistant to save the event shown on the screen to the calendar app, or an integrated calendar feature inside the app itself. Because the feasibility of implementing an integrated calendar into the app was questioned, therefor the event shall be imported into one of the existing calendar on the users device.

- ***Booked events query***

The idea is to enable the user to ask the built-in assistant about the users personal schedule, the user may say “when is my next event?” and then the assistant shows the next event description screen if any were found (see figure 21 for general steps required).

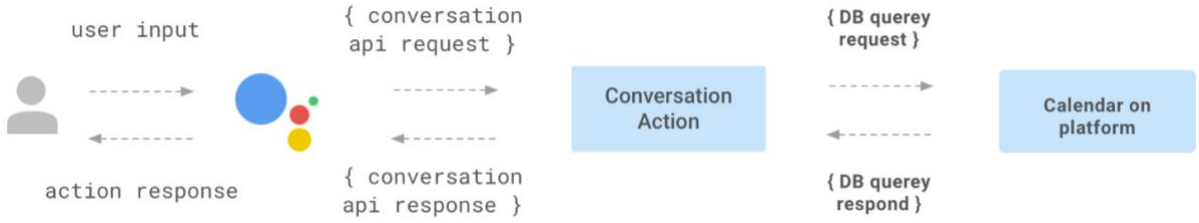


Figure 21 Booked events query

4.4 Mock-ups

Prototyping and creating Mock-ups was the key of reaching a state of the design, that assures all parties, that this app will deliver a high level of user experience design and usability, as well as finding out the flaws in design at early stages before changing them become costly.

- ***Get inspirational designs***

"Good artists copy, great artists steal"[47], the team interpretation of this quote is that the design should not copy the other designs one to one, but rather follow their steps to come up with a design that fulfils the clients requirements and the end-users needs.

Thus an intensive research has been made to create an inspirational board, which to be found in the appendix, and some sketches were done via papers to see how the sample of the end-users would react towards those suggested designs.

- ***Paper prototype (Low-Fi)***

The paper prototype was meant to be use as a guidance on how to navigate through the screens, and where to place the features, buttons and represent the app major and minor functions.

The app home screen was first designed by placing all the major features on the home page (figure 22). The reason for that is to enable the end-user to start their tasks fast, accordingly reduce the time required to reach goals.

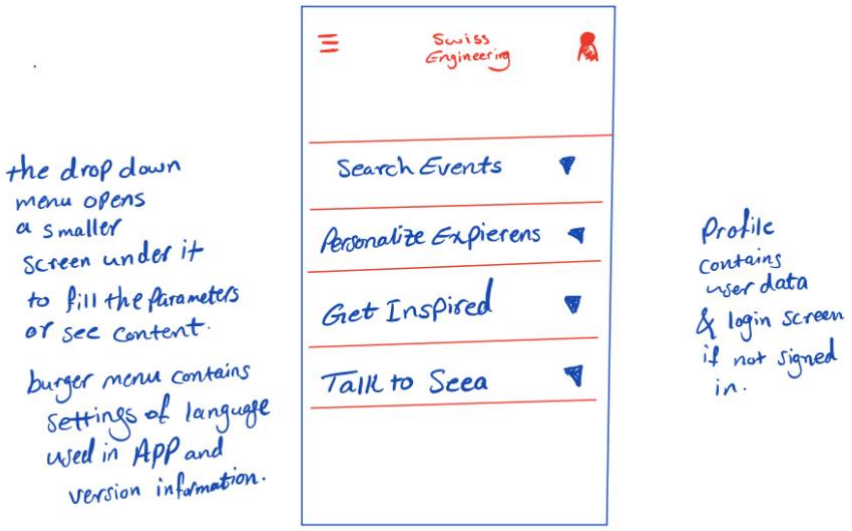


Figure 22 Home screen first concept

The idea of the first design is to include all the elements into one screen, and use drop down views to enter all required inputs. The design story in details can be found in the appendix.

The problem of the first design according to the end-users feedback is that all elements are compacted into one single screen, so at least two taps are required for each input (first tap to open the drop-down menu, then the second to focus on the input field and show the keyboard).

Additionally, irrelevant elements will still be visible on the same screen, although the end-user's intent is to do a different task. For instance the Search Event tab at the top will still be visible even though the end-user taps on the Personalize Experience tab below it. Overall the end-users did not like this implementation for the Home page, thus the new design was implemented instead.

Another important aspect is navigating through the screens to enter inputs. The idea of navigation through "Search Event" screens for instance was to have a specific information entry screen for each attribute, and the number of steps needed (screens to come) is represented by the empty dots at the top (see figure 23), where the filled dot is the location of the current screen in the queue.

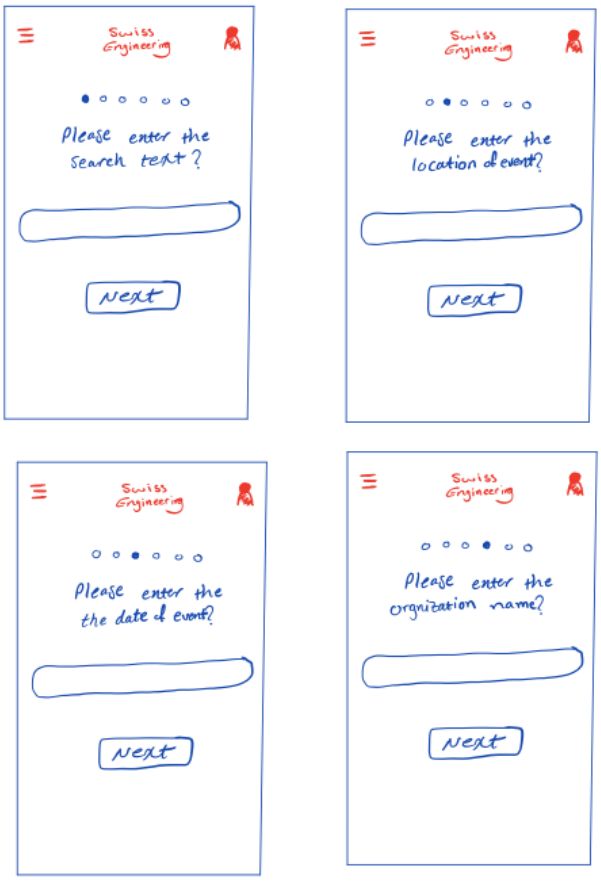


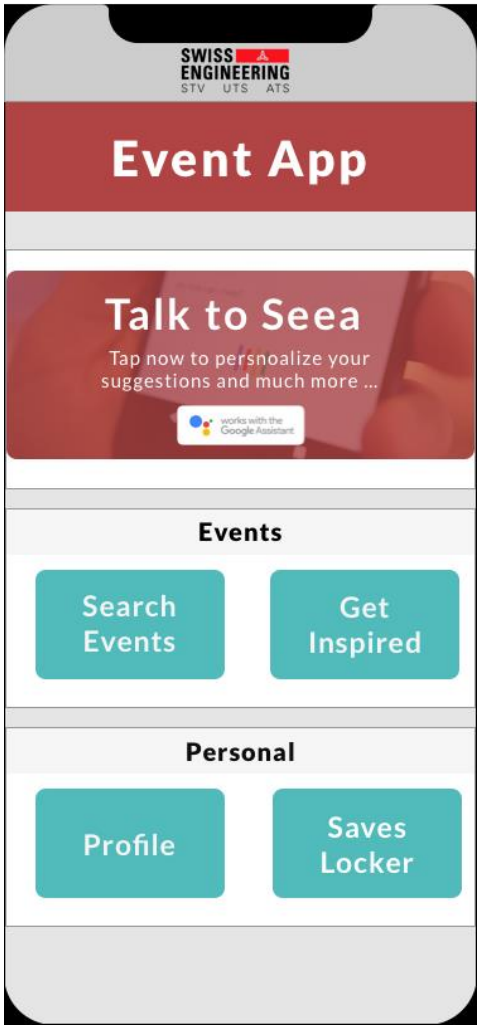
Figure 23 gathering the user input in steps

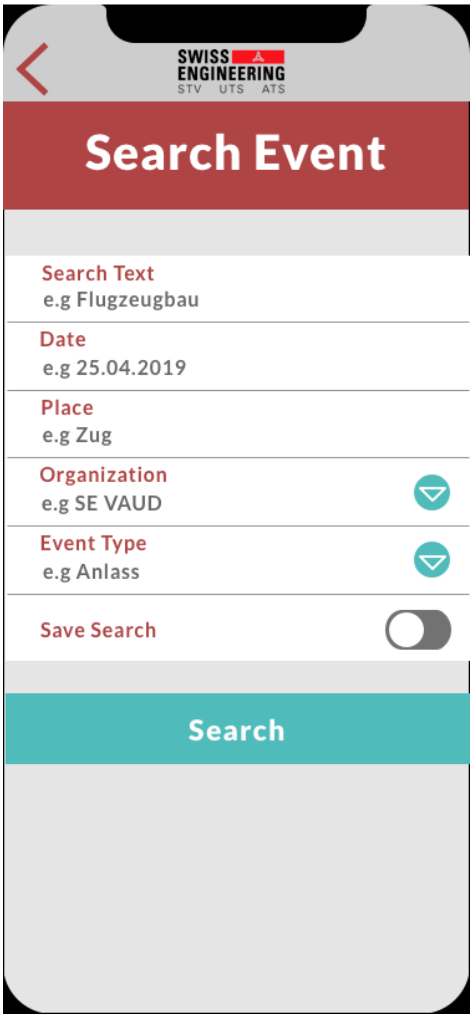
A major problem of this design is that in order to view the list of all events, with no search criteria applied, the end-user must tap 7 times on the next button without entering any data to reach the goal.

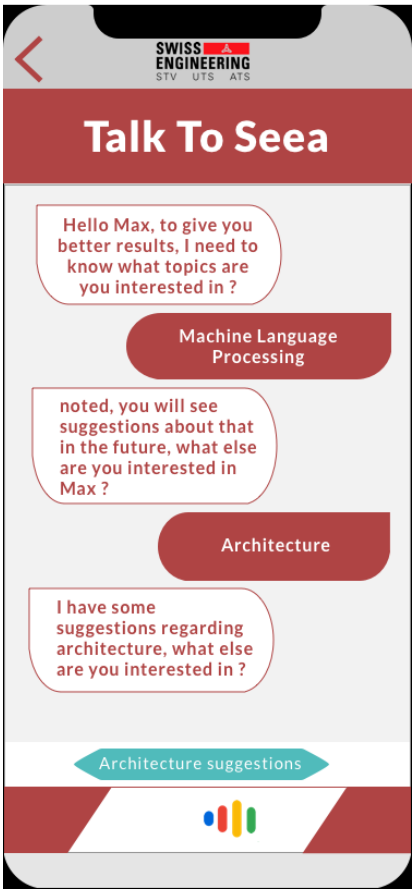
The solution was therefore to adapt the classical structure of input entry screen, which is a single screen to collect all the criteria to apply for a search query. A full list of experimental screens can be found down in the appendix.

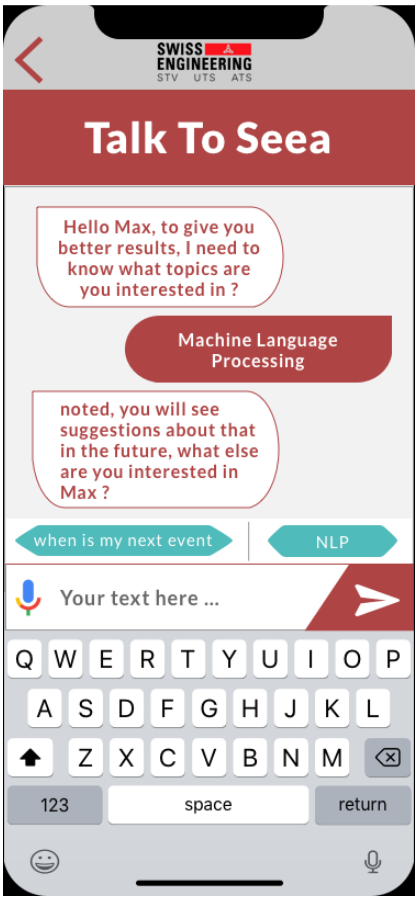
- ***Hi-Fi Prototype (Sketch Project)***

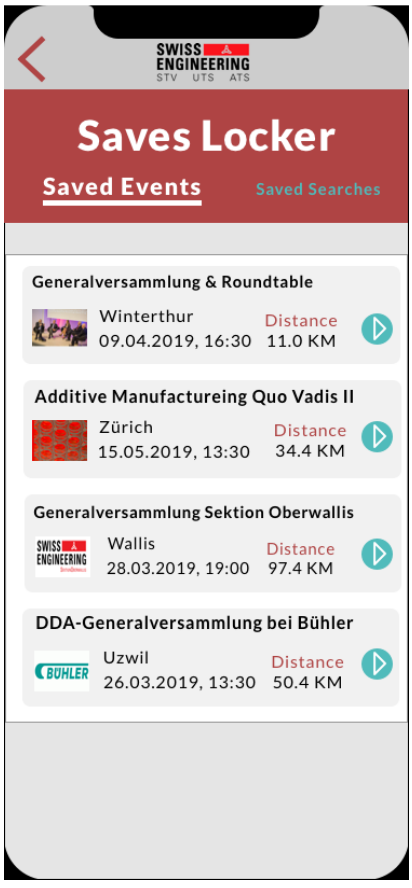
The following Hi-Fi prototype screens (see figure 24) was done to assure that every UI element, color and user interaction satisfies both the end-users and client's needs. The full story of the Hi-Fi prototype evolvement is included in the appendix.

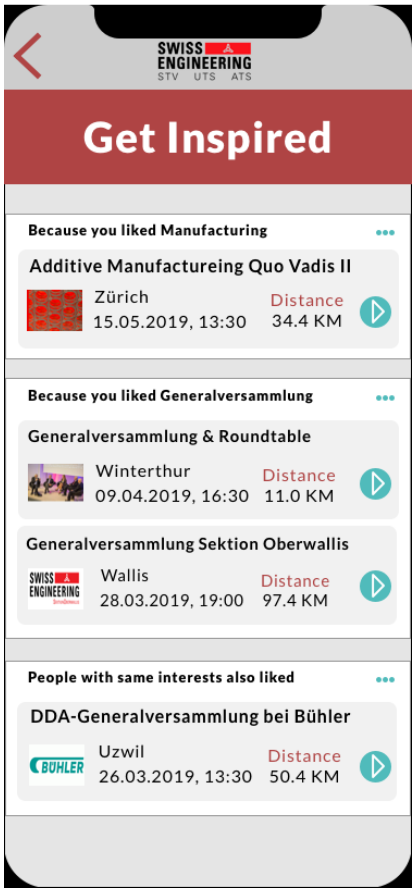












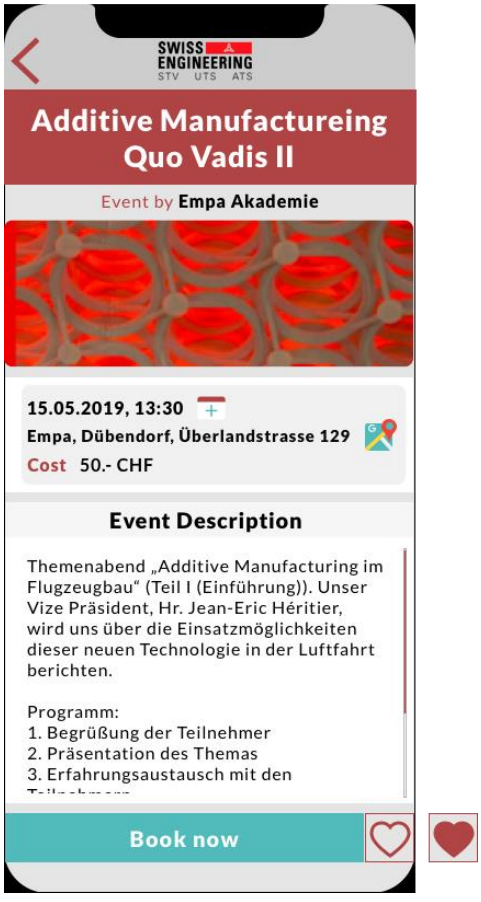


Figure 24 Main Hi-Fi Sketches screens

5. Implementation

This chapter will discuss the technical implementation of the Swiss Engineering App prototype.

All the limitations will be presented, as well as the architectural design of the prototype and the code implementation.

All the important used external plugins would be addressed as well to illustrate their roles.

5.1 General Decisions and Limitations

Based on the technical research conducted, the following decisions have been made.

- As the scope of this prototype is to prove the feasibility of implementing a voice feature inside the event app, the decision has been made to implement the voice feature using a Text To Speech plugin. All plugins researched can be found in the research chapter.
- According to end-user's interviews results presented briefly in the requirements sub-chapter, the scope of using the voice feature will be limited in this prototype to enter the tags name, therefore no chatbot has been linked with this prototype, instead the tag extraction process is done using Regular Expressions implemented inside the code. However, it is recommended to implement a chatbot in case the voice feature is expected to provide more functionality in the future.
- Google Assistant Actions have not been used, as at the moment it enables the action to work inside the Google Assistant app, and not inside another third party application.
- No online connection has been established with the Swiss Engineering Database. Unfortunately the model designed in this prototype and the information to be displayed are not well organized in the Swiss Engineering Database (e.g the dates do not have their own attributes, and the addresses are not well structurally written). Thus an array of events has been created in code and has been used as the data source instead.
- Booking events is not possible using this prototype. The reason is the lack of consistency when it comes to book events on the Swiss Engineering platform, as most of the events are not booked via a button click, rather via mail contact with the organizers.
- Get Inspired page has not been implemented due to the lack of man power as the team consists of only one member.

5.2 Technologies used

This sub-chapter will discuss all the details regarding the technologies that has been adopted inside this prototype in order to deliver the intended functionality, as well as the reason(s) to choose those.

5.2.1 Developing Platform

As it was mentioned in the project description, the goal of this project is to develop a native application for both Android and iOS, thus Xamarin for Visual Studio (v8.2.2 Build 21 for mac) was the chosen development platform.

Visual Studio has also a XAML Previewer[48] integrated into the platform, which make it easier for developers to see the changes in design while writing XAML code without running the project. Collectively, that should also reduce a substantial amount of time.

The number of tutorials available online for both C-sharp programming language and Xamarin.Forms (v3.6.0.344457 for mac) was a major factor of choosing Xamarin over the other less-known programming languages and development platforms listed in the research chapter.

Xamarin solution consists normally of one shared project, which contain the code for all operating systems, and then one project for each operating system the application should support. In this prototype for instance, the Xamarin solution consists of one shared project for both iOS and Android, one project specifically for iOS and another one for Android, which make them 3 projects in one.

Xamarin is the only researched platform, that offered the opportunity to write code in Swift or Java for the iOS and Android projects, thus more customization options and less limitations.

Those platform-specific projects are part of any solution initialized by Xamarin in addition to the shared project, which is written using C-sharp code, and they are available in case more customization is needed than offered by the shared project.

If no platform-specific customization is needed, then those specific-platform projects can be left without change, which was not the case in this prototype.

The existence of instructors at the FHNW who have an outstanding knowledge in both C-sharp and Visual Studio was also an important factor of this decision.

Other than that, the Xamarin developers society is very helpful, and they tend to answer the questions in a short period of time.

Visual Studio is also a well-known developing platform for the team. The team members are used to write code in Visual Studio as well as do debug.

5.2.2 Plugins

This sub-chapter will discuss the most relevant plugins that has been adopted inside the project and their usage.

- ***Xam.Plugins.TextToSpeech***

This plugin has been used to provide the Text To Speech functionality.

Xam.Plugins.TextToSpeech plugin is the one with the highest number of downloads as presented in the research chapter, other than that it has good reviews online among the Xamarin Developers society.

The plugin works well with the current version of Xamarin for Visual Studio (v8.2.2 build 21), and provides stable results, although it was integrated in the project since the previous version of Xamarin for Visual Studio.

Using this plugin and integrating it into the app is fairly simple, and there exist many online tutorials to demonstrate this process.

- ***Xam.plugins.Forms.ImageCircle 3.0.0.5***

This plugin is used as a custom control for Xamarin.Forms project to turn images into circle images. It offers also customizable border, thickness and color. [49]

- ***Rg.Plugins.Popup 1.1.5.188***

This plugin is for Xamarin forms. It allows to open any page as a popup view. [50]

There exist many tutorials online demonstrating how to integrate this plugin into an existing project. It provides stable results while functioning the way it should.

- ***Plugin.Persmissions 3.0.0.12***

Simple cross platform plugin to request and check permissions. [51]

This plugin manages the access permissions to the apps and components on the user's device (e.g Camera app) as well as some data that must be previewed in order to make user's decision on this prototype (e.g access the calendar data available on the device to preview the existing calendars names).

- ***CClarke.Plugin.Calendars 1.0.0***

This plugin Perform basic CRUD calendar operations (Create, Read, Update and Delete) on iOS/Android/UWP through a simple common API. [52]

This plugin has been used in this prototype to interact with the end-user's calendars available on the device, so that an event can be imported into a calendar chosen by the end-user.

5.2.3 Speech To Text

The Speech To Text (STT) feature is required to convert the voice input entered by the end user to a textual representation, in order for the tag to be extracted. More on that comes in details in the following sub-chapters.

As the Text To Speech feature was implemented using the external plugin presented in the previous sub-chapter, the Speech To Text has to be also implemented independently. Thus is the existing Speech To Text API on the device is used.

Android has implemented its own Speech To Text feature on all its Android devices, the same applies to the iOS devices in which Apple's own Speech To Text feature is implemented as an accessibility feature.

Those two APIs are used in this prototype in order to convert the user's speech into text, for both Android and iOS.

The advantage is that no external plugin is required to implement either of those two as they are already built-in inside the device's operating system, and the operating system's companies are responsible of managing and maintaining those as they are an essential part of their operating systems (accessibility feature).

To integrate those two APIs, an Interface called "ISpeechToText" has been created, which is then implemented inside the platform-specific projects.

The implementation of this interface is a class called "SpeechToTextImplementation", which exists in both Android and iOS projects.

5.3 Architecture

The application consist of 5 features packages (see figure 25), where the home page is the start point to interact with each of those features.

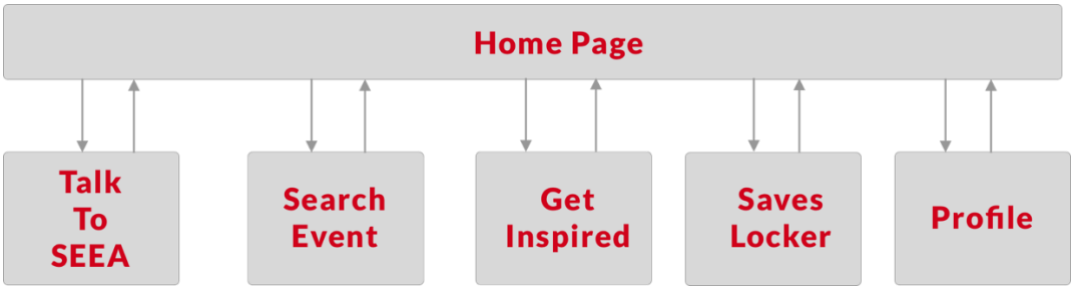


Figure 25 Solution's main packages

Each of the packages listed in figure 25 consists of a sequence of screens, where each screen checks the input if necessary and then passes input to the next.

Profile and Get Inspired are implemented each as screens to display information. Saves Locker is designed as a Tabbed Page to contain the saved searches as well as the saved events, each displayed in its own separate screen, the end-user can navigate between both using the navigation bar implemented at the bottom.

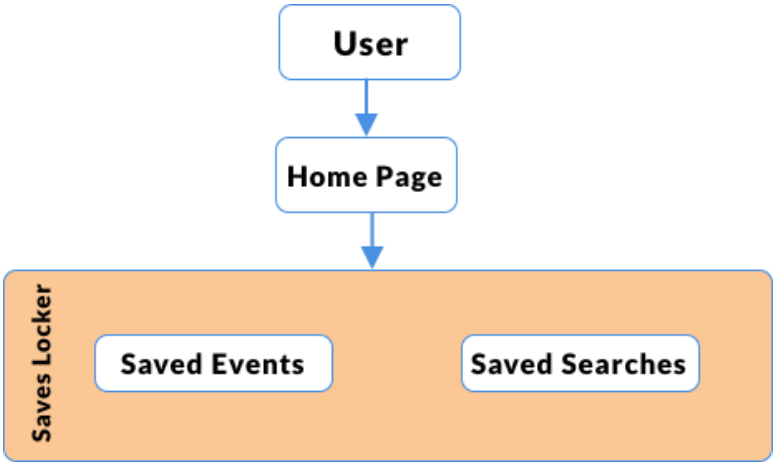


Figure 26 Saves Locker Tabbed Page with Saved Event and Saved Searches implemented separately inside

- ***Talk To SEEA***

Talk To SEEA handles two types of input, textual and voice input.

Where textual input is handled using Regular Expression to extract the tag name, the voice input must be first converted into text.

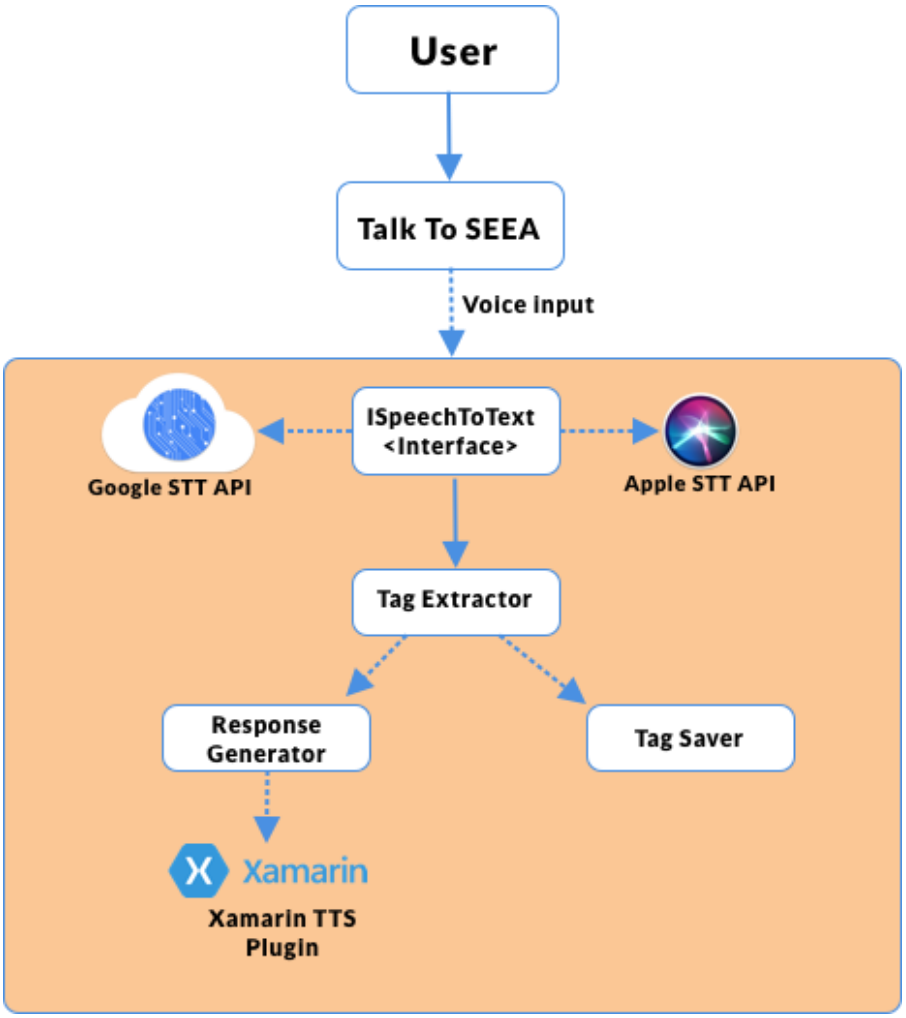


Figure 27 Talk To SEEA design architecture

As illustrated in figure 27, the voice input is converted to text using the device existing Speech To Text (STT) API, either for Android or iOS devices. Where the textual input will go directly into the Tag Extraction phase.

There exist two classes called SpeechToTextImplementation in both iOS and Android projects, those classes implements the ISpeechToText interface. More details can be found in the code part.

The response generated is being spoken using the Xamarin TTS plugin in addition to a written text displayed on the user's screen, if the user's input was entered via voice. Otherwise the response will be just displayed as text to be read.

- ***Search Events***

Search for events is a sequentially ordered set of screens in order to get the user's search criteria and display the result. The functionality and set of constrains applied on those inputs is illustrated more in details in the next sub-chapter.

The event description page on the other hand uses multiple plugins in order to enable the user to import the event details into the user's calendar of choice (see figure 28).

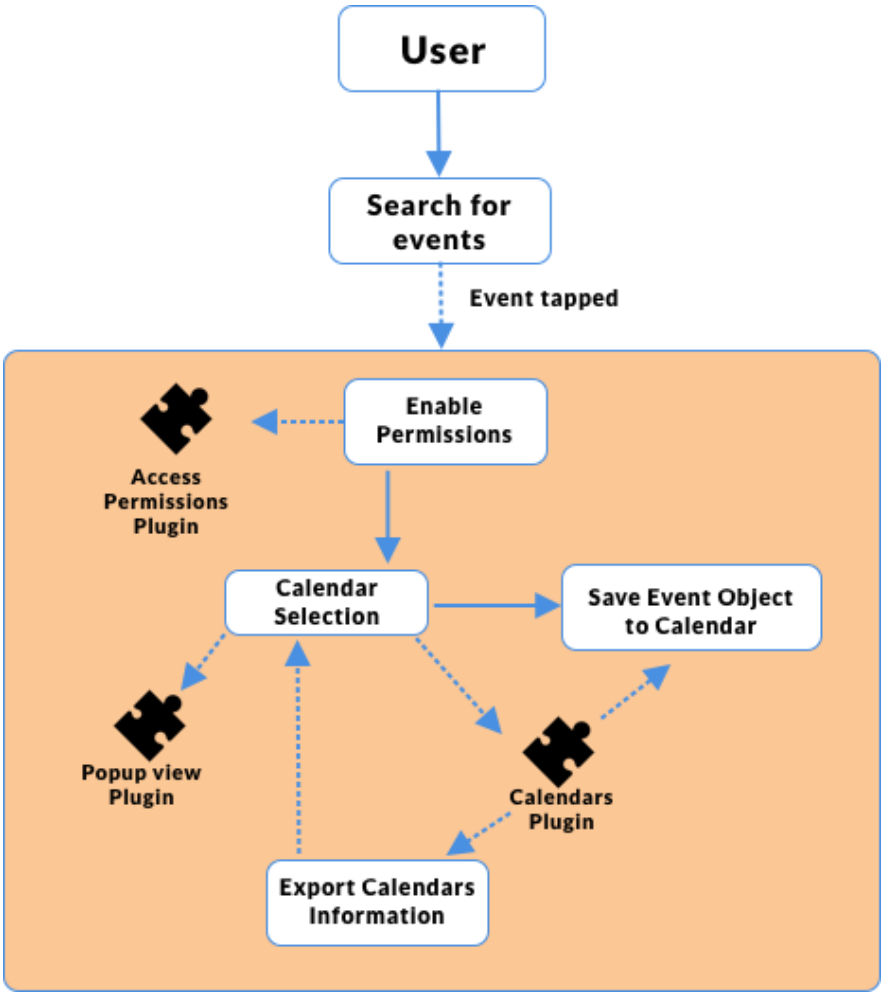


Figure 28 Export Event to Calendar

The Permissions to access the calendars are managed and granted by the Permissions plugin.

The Popup plugin view is used to display the list of available calendars loaded on the device after exporting their information using the Calendars plugin.

The Calendars plugin is used as well to create the Event's object in order to export it to the calendar chosen by the end-user previously.

5.4 Preview Logical Functionality

This sub-chapter discuss and previews the logical part behind some of the main functions that were implemented in this prototype.

As Search For Events, Talk To SEEA and Get Inspired features have a bit complicated logic than the Profile and Saves Locker screen, the following is a more in depth logical explanation for those.

5.4.1 Search for Events

The first screen in the search for events feature asks the end-user to enter the search criteria.

Those search criteria (see figure 24). can either be left empty to preview a list of all available events to book, or filled with search criteria of preference (see figure 29).

The search function creates a new list, which is the result events list, then iterates through the whole events existing and add to the empty list what matches the user entered criteria (see figure 30).

If a date was entered by the end-user, it must fulfil some conditions.

The date must be entered in the form of (dd.mm.yyyy) and can not be more than 5 years in the future.

The date can not contain any characters.

The date must consist of only numbers, no characters are allowed. And it must not be in the past.

The day must be also between 1 and 31, and month between 1 and 12.

To see the logical operations done to check on the date more on details please refer to figure 31.

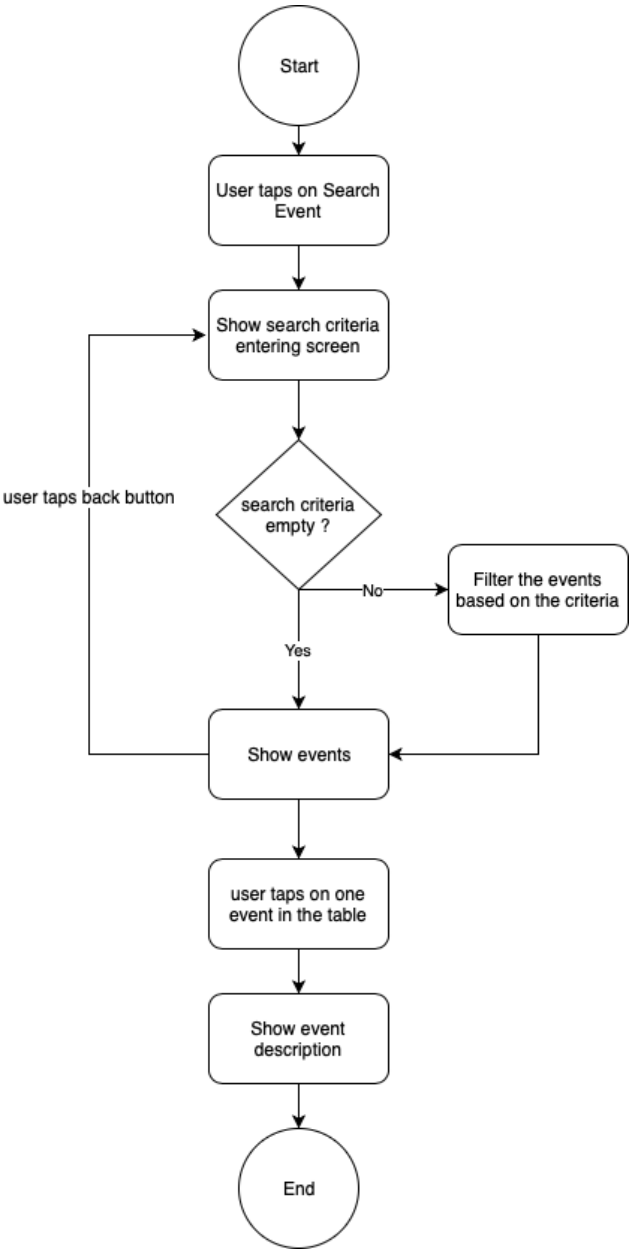


Figure 29 Search for event overall process

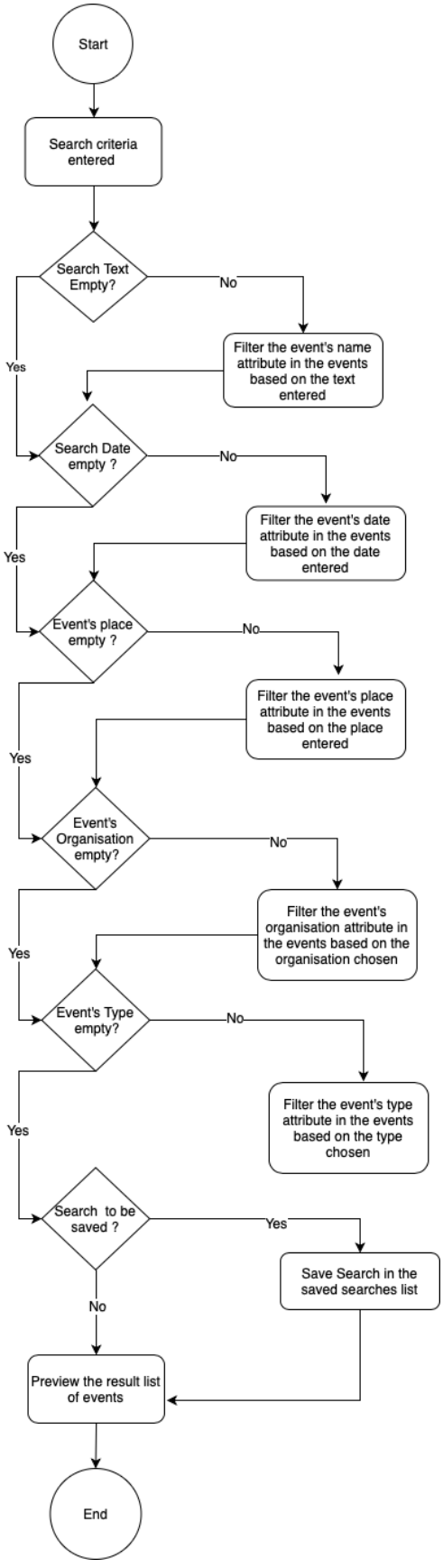


Figure 30 Filter results based on criteria

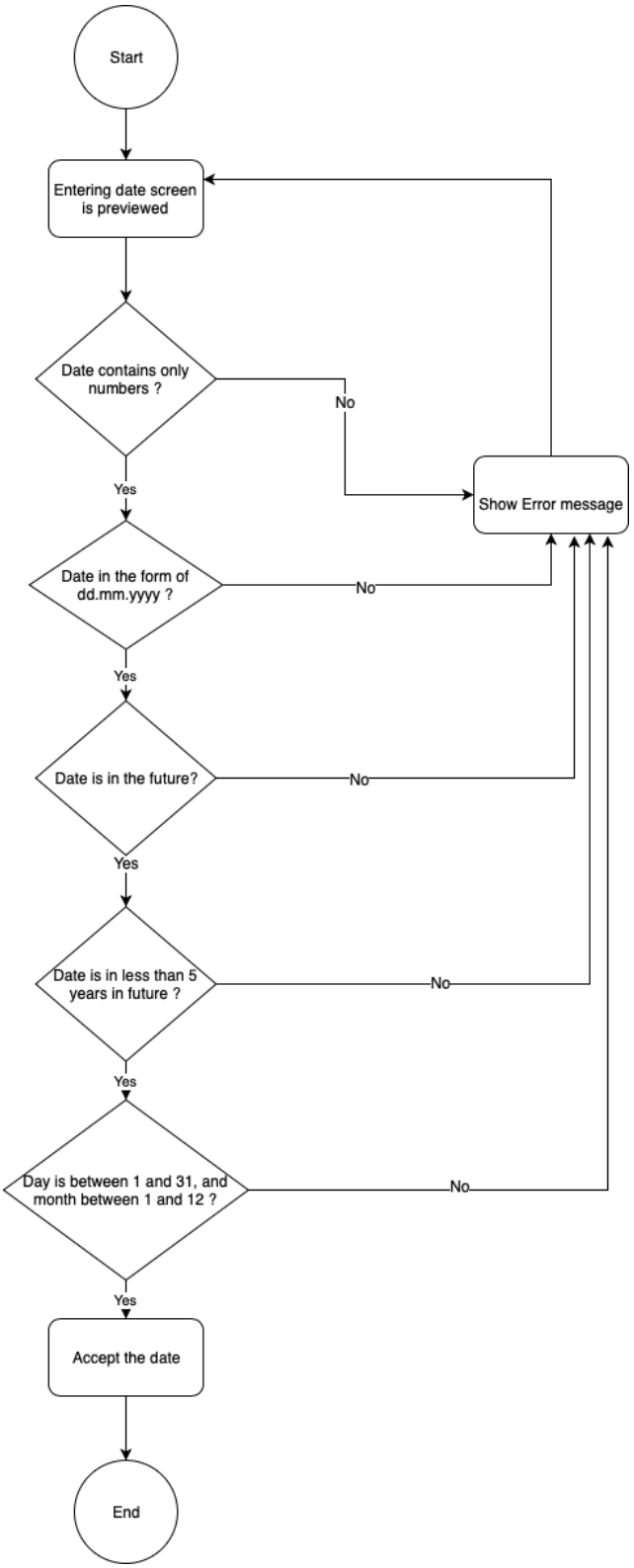


Figure 31 Decide whether or not a date is accepted

5.4.2 Talk To SEEA

Talk To SEEA requires a specific set of words to appear in the users input in order for it to function correctly and extract the user's tag from the sentence.

The idea behind Talk To SEEA (see figure 32) is that it detects what comes after a specific set of words, namely "I am interested in <tag>", "I love <tag>" or "I like <tag>" where tag is the name of the topic the user is interested in and wishes to receive suggestions about in the future.

It works as it will be illustrated in details in the Code Implementation sub-chapter using the regular expression (REGEX), which helps to detect the appearance of the trigger words and extracts the tag name that comes afterwards.

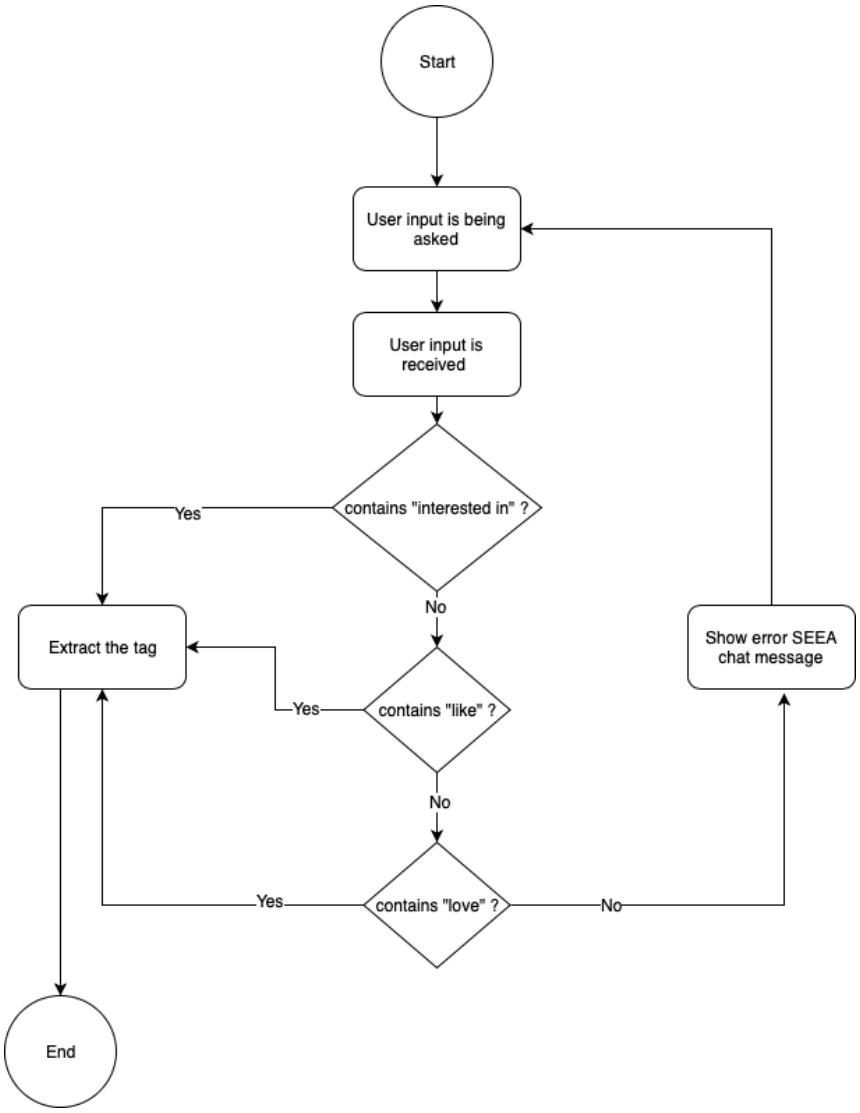


Figure 32 Checking the existence of the trigger words in the user's input

If the user's input was detected via voice, then this voice input must be converted into a textual representation first in order for the tag to be extracted, otherwise the tag extraction process started directly.

The tag detected must be then saved into the end-user's tags preference list.

5.4.3 Get Inspired

Get Inspired screen previews the events that have been suggested by the simple Artificial Intelligence implemented to the end-user.

The events that matches either the user's tags of preference or the Social Factor illustrated before are being suggested in this screen.

By opening this screen, the application scans the list of the available events that contains tags which matches the user's liked tags.

Additionally, the events that are booked by a former co-attendees are also shown in this screen. Which requires to keep track of the former co-attendees yet anonymously, so that their privacy is not violated.

Figure 33 describes the process of gathering suggestions sequentially.

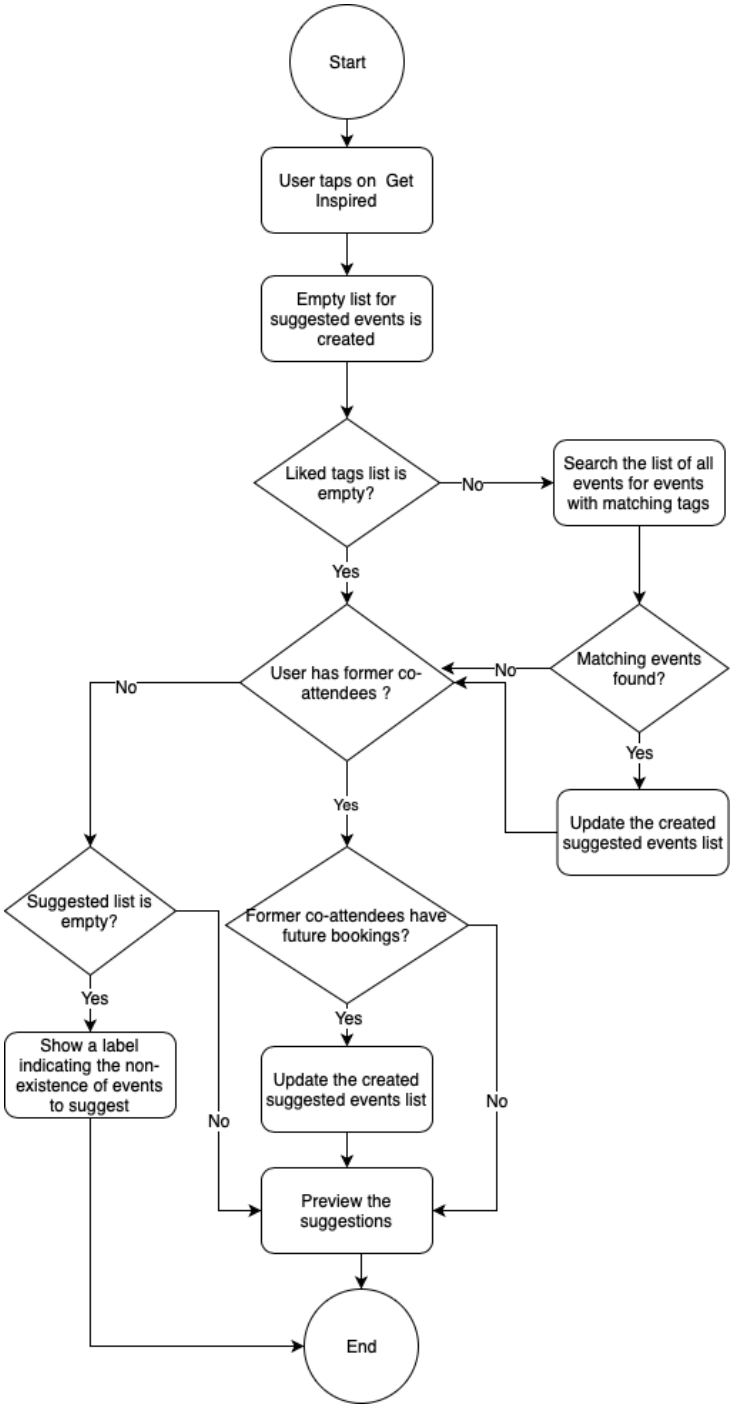


Figure 33 Get Inspired screen logical operations

5.5 In-App Screenshots

This sub-chapter previews the in-app screenshots of the main implemented screens, while others which contain the warning labels are left to be seen inside the application itself.

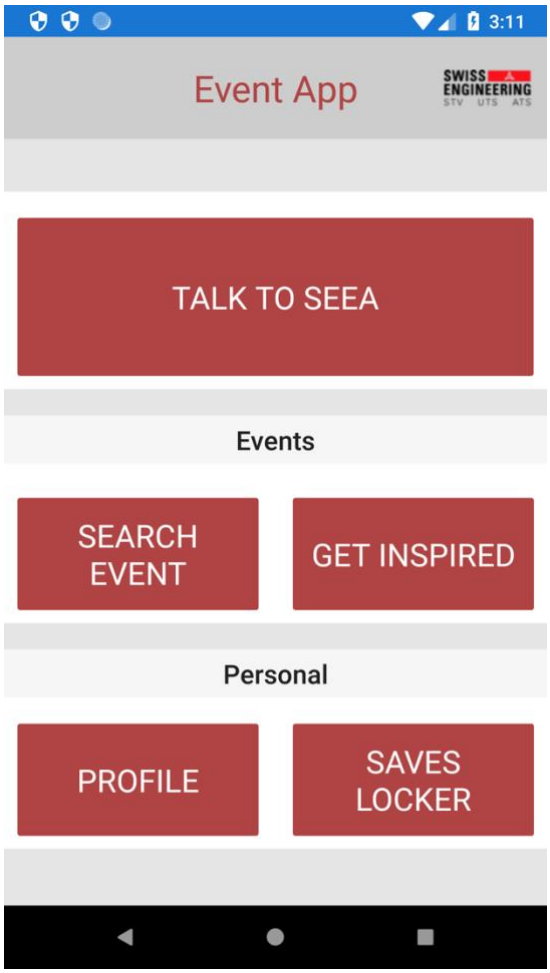


Figure 34 Application's home screen

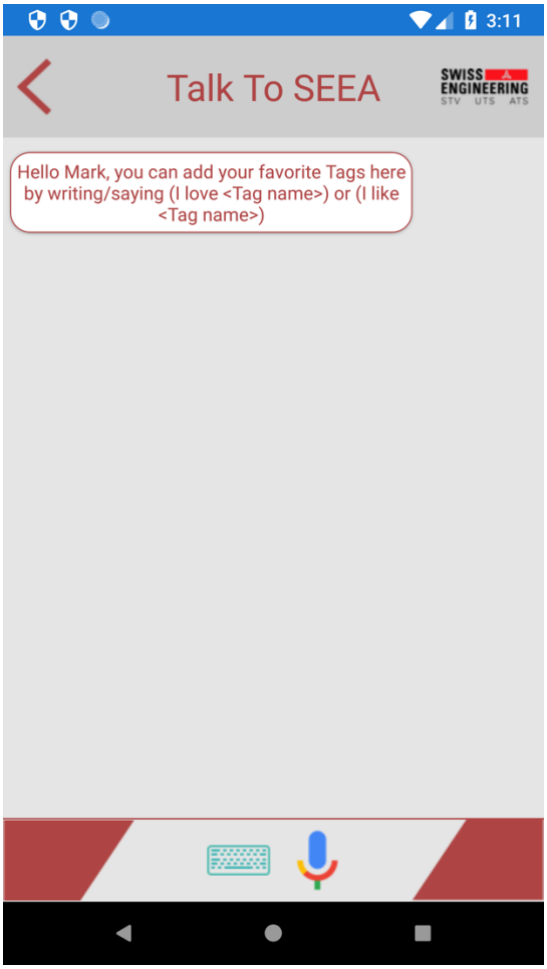


Figure 35 Talk To SEEA not listening mod

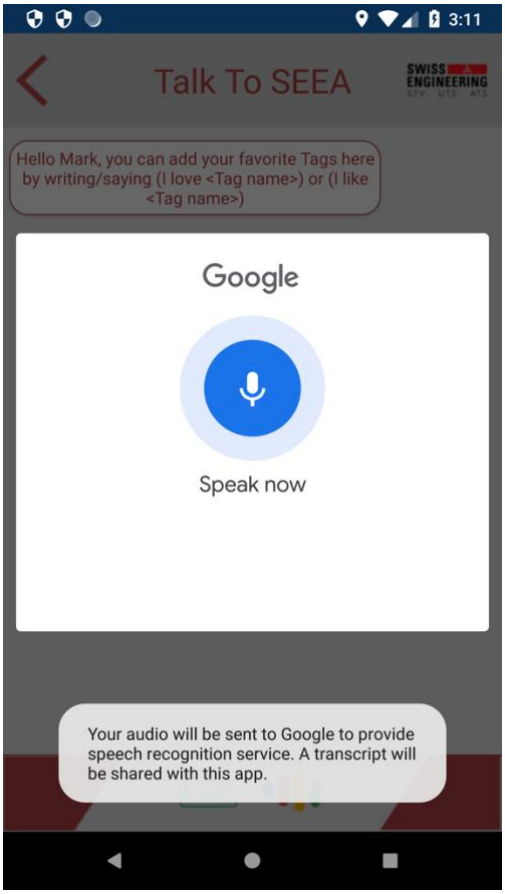


Figure 36 Talk To SEEA listening mode

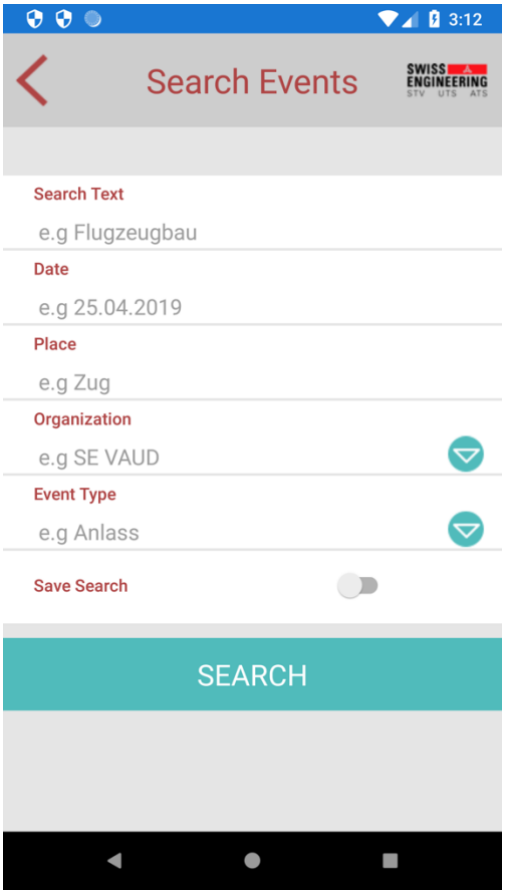


Figure 37 Search for events entry screen

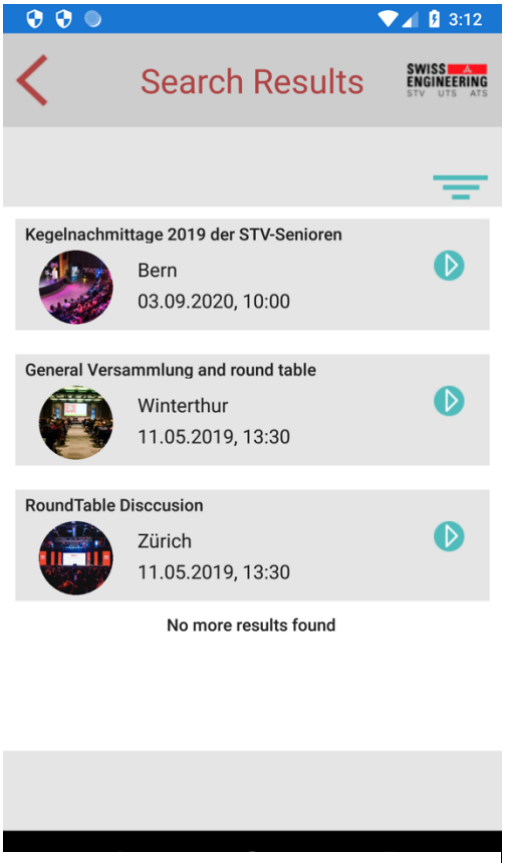


Figure 38 Search results

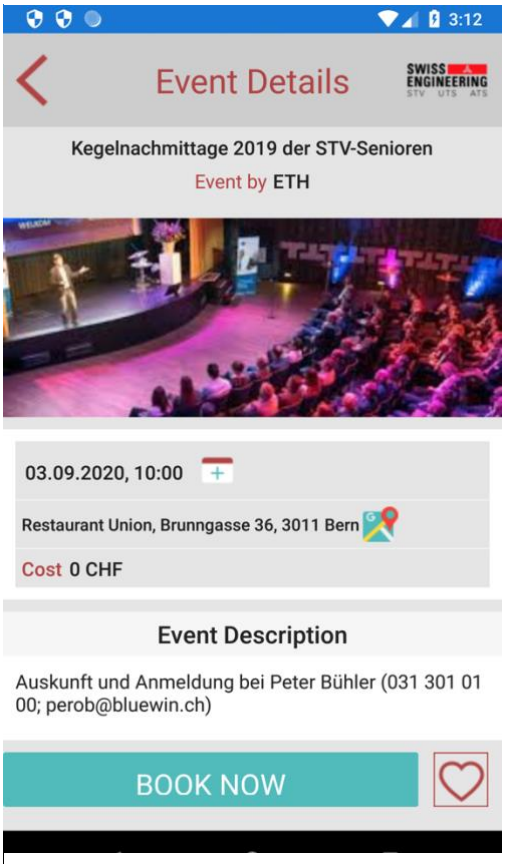


Figure 39 Events details screen

Figure 40 Profile main screen

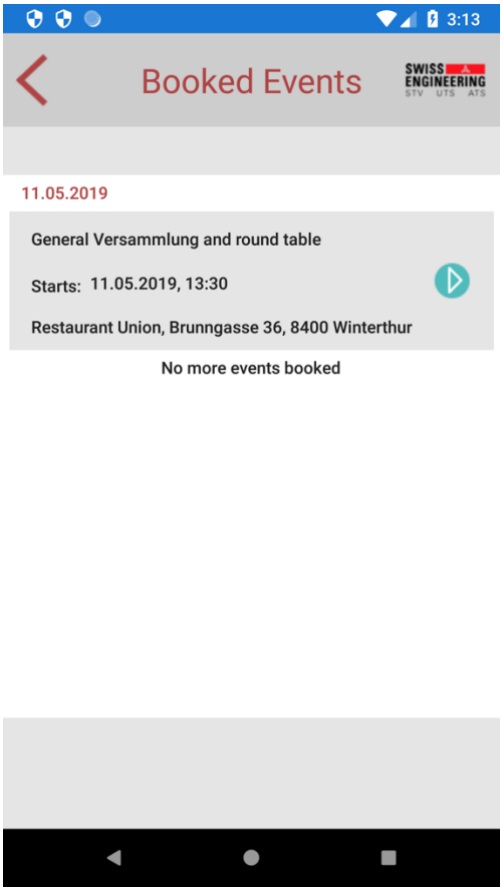


Figure 41 Booked events screen

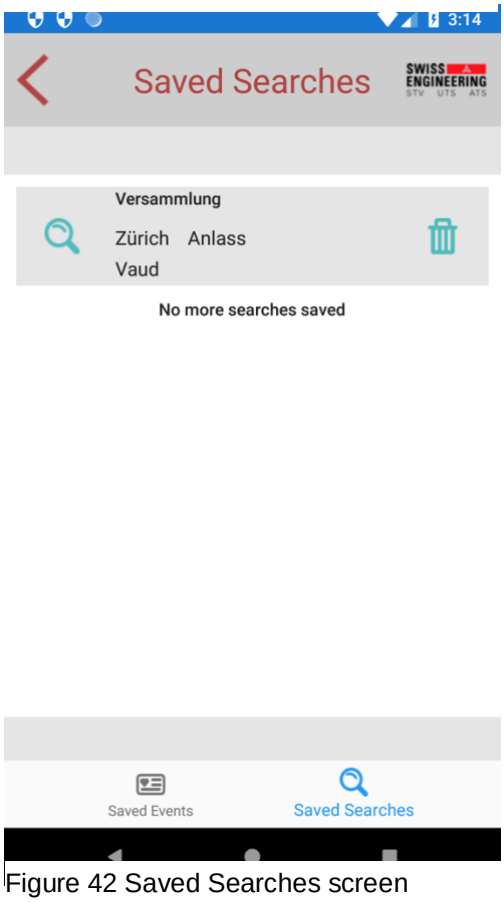


Figure 42 Saved Searches screen

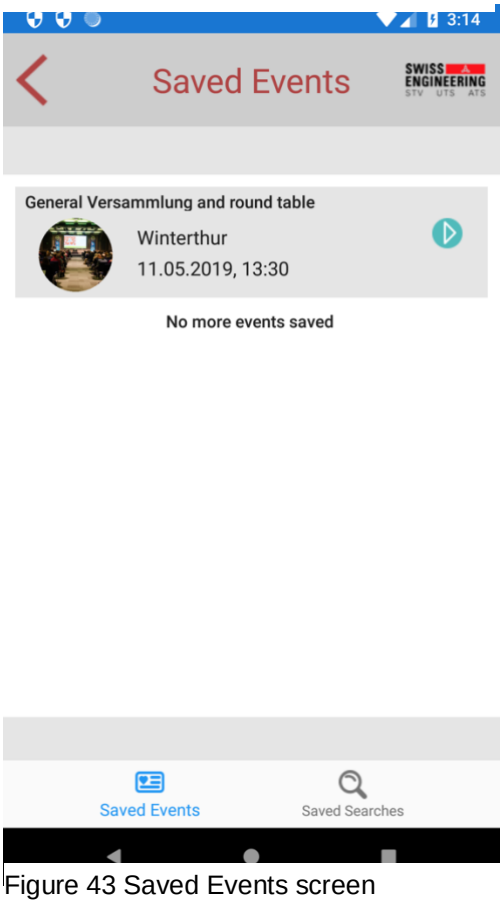


Figure 43 Saved Events screen

5.6 Code Implementation

The previously described functionality is implemented in the classes contained in figure 44.

The main classes are contained under the folder with the name of the application's features.

While the others are either customization or objects classes, that are in use and referenced in the main classes.

This sub-chapter discuss the code implementation of the previously discussed features and functions of the Swiss Engineering Event prototype application.

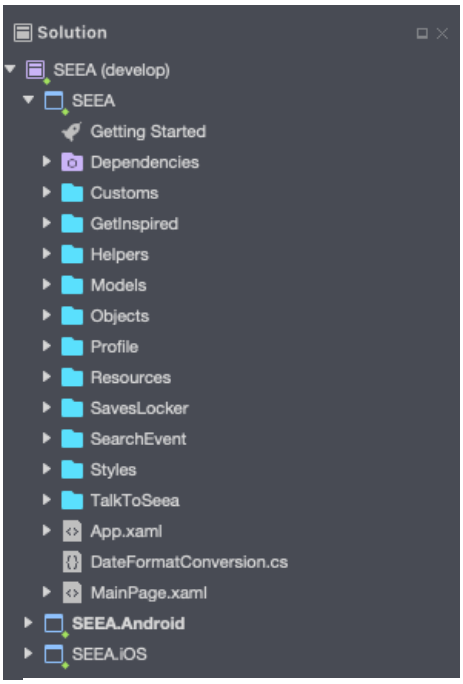


Figure 44 Project code folders

5.6.1 Talk To SEEA

Talk To SEEA has one screen that shows the interaction between the end-user and the simple AI called SEEA.

The screen has a stacklayout object, that contains those interactions ordered chronologically, where the most recent comes at the end.

```
private void add_new_seea_msg(String msg) {

//construct a SEEA chat frame with the correct colors.
SeeaTalkFrame new_msg = new SeeaTalkFrame(Seea_talk_bg_color,
Seea_talk_text_color, Seea_talk_border_color);

//set the response inside the frame constructed.
new_msg.Text = msg;

//Add the frame to the stacklayout.
chat_messages_stack.Children.Add(new_msg);

//scroll to the end of the screen to include the added message (frame) in
the scope.
chat_message_scrollview.ScrollToAsync(new_msg, ScrollToPosition.End,
false);
}
```

Code snippet 1 add message to SEEA chat

This stacklayout contains both messages of SEEA and the user, where the color of each helps to differentiate.

This detection (as illustrated in the previous sub-chapter) is done via Contains() and After() methods applied on strings where both textual or voice input is converted into string first, then using REGEX this string is trimmed so that only what comes after the trigger words is left.

This detected tag is then being saved and compared to the events to see which contains this tag, so that that event is being suggested to the user and previewed in the Get Inspired page.

Handling users input

Talk To SEEA interaction has two different models. The following sub-chapters will discuss briefly how the users input is being handled and parsed into a string so that the tag can be extracted form.

Figure 45 Entering the users input via text (Mock-up screen)

- ***Interacting via text***

The user in this mode must enter the tag name as illustrated previously via text using the virtual keyboard (see figure 45).

In this mode there is no need to use any extra plugin to convert the input, as it is already a string (text).

- ***Interacting via voice***

The user in this mode must enter the input via voice (see figure 35). The speech must contain one of the mentioned above forms in order for SEEA to be able to extract the tag.

The user is expected to tap on the mic button in order for SEEA to start listening, then while listening the mic image shall change to indicate that SEEA is now listening (see figure 36).

This switching between figures shall provide an extra layer of privacy to the user, as SEEA is only listening when the user sees the listening figure on the screen.

In order to provide the ability to capture the users speech, the existing Speech-To-Text provided on the mobile device is being used (see figure 36). Therefor are two classes "**SpeechToTextImplementation.xaml.cs**" implemented, one in each project for each platform – iOS and Android – to use the built-in Speech-To-Text APIs.

The method StartRecordingAndRecognizing() is the responsible of capturing the users speech, and also handling the cases where the speech is a complete silence for instance.

```
try
                {
var voiceIntent = new Intent(RecognizerIntent.ActionRecognizeSpeech);

voiceIntent.PutExtra(RecognizerIntent.ExtraLanguageModel,
RecognizerIntent.LanguageModelFreeForm);

voiceIntent.PutExtra(RecognizerIntent.ExtraPrompt, "Speak now");


voiceIntent.PutExtra(RecognizerIntent.ExtraSpeechInputCompleteSilenceLength
Millis, 1500);

voiceIntent.PutExtra(RecognizerIntent.ExtraSpeechInputPossiblyCompleteSilen
ceLengthMillis, 1500);

voiceIntent.PutExtra(RecognizerIntent.ExtraSpeechInputMinimumLengthMillis,
15000);
voiceIntent.PutExtra(RecognizerIntent.ExtraMaxResults, 1);
voiceIntent.PutExtra(RecognizerIntent.ExtraLanguage,
Java.Util.Locale.Default);
_activity.StartActivityForResult(voiceIntent, VOICE);

                }
```

Code snippet 2 Android Speech-To-Text API usage in class SpeechToTextImplementation.cs

iOS implementation on the other side is more complex, as it creates an instance of AVAudioSession to record the users speech, and then uses the SFSpeechRecognizer to create what is called a transcript of the users speech.

```
_audioEngine.Prepare();
            _audioEngine.StartAndReturnError(out nsError);

            _recognitionTask =
_speechRecognizer.GetRecognitionTask(_recognitionRequest,
                                     (result, error) =>
         {
          var isFinal = false;
          if (result != null){
              _recognizedString = result.BestTranscription.FormattedString;
              MessagingCenter.Send<ISpeechToText, string>(this, "STT",
                                                          _recognizedString);
              _timer.Invalidate();
              _timer = null;
              _timer = NSTimer.CreateRepeatingScheduledTimer(2, delegate
                     {
                         DidFinishTalk();
                     });
           }
           if (error != null || isFinal){
             MessagingCenter.Send<ISpeechToText>(this, "Final");
             StopRecordingAndRecognition(audioSession);
           }
   });
```

Code snippet 3 a part of the implementation of StartRecordingAndRecognizing inside the SpeechToTextImplementation.cs for iOS project

ISpeechToText interface has two important functions which are StartSpeechToText and StopSpeechToText which are implemented inside the SpeechToTextImplementation.cs classes in both projects.

This interface is used inside the portable project to call the correct implementation of the class SpeechToTextImplementation.cs whether it was the iOS or Android verison depending on the mobile device which the app runs on.

```
try
            {
    speechRecongnitionInstance = DependencyService.Get<ISpeechToText>();
            }
catch (Exception ex)
             {
    Log.Warning(ex.Message, "");
            }
```

Code snippet 4 calling the correct class that implements ISpeechToText interface

The users input can be then accessed in the portable project using a list that contains the users sentence called inputs.

```
List<String> inputs = new List<String>();
String platform = Device.RuntimePlatform;
```

Code Snippet 5 the inputs list inside the portable project

The list has in each index the so far detected sentence. Because of that are the last two indexes duplicated (see figure 46), as the previous before the last contains the sentence with the last word detected, and the last one contains the full sentence detected.

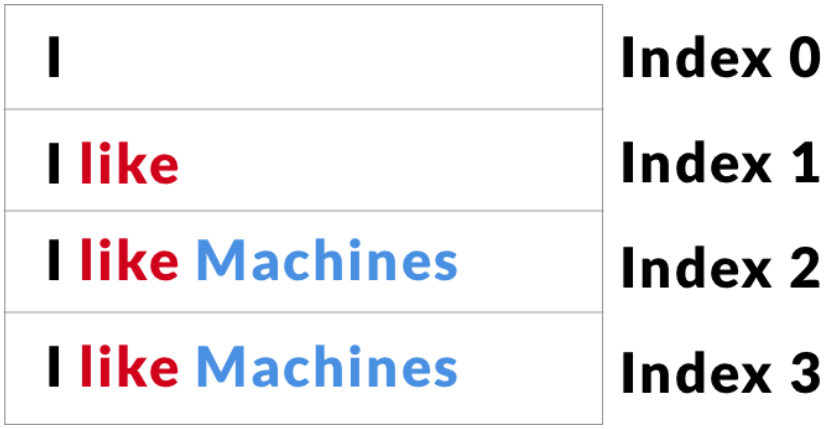


Figure 46 how are the words captured inside the inputs list

Therefor SEEA shall wait till the last two indexes are equal, then take the last one and handle it as the actual users input.

```
if ((inputs.Count != 0))
                {
   if (inputs.ElementAt(inputs.Count - 1) ==
       inputs.ElementAt(inputs.Count- 2))
                    {
       if (platform == "iOS")
                        {
           mic_btn.Source = "googleMic.png";
           add_new_user_msg(inputs.ElementAt(inputs.Count - 1));
           TriggerSeeaRespond(inputs.ElementAt(inputs.Count - 1));
           inputs = new List<String>();
                        }
                    }
                }
```

Code snippet 6 comparing the last two indexes inside the inputs list

Extracting the tag from the users input

The Rule-Based Chatbot's rules can be found in the Chatbot class, which is called inside the TalkToSeeaScreen class using an initiated object.

Extracting the tags – topics names – is done using Contains() and After() methods applied on the stringified user input. A set of predefined sentences must be said in order for the simple AI behind SEEA to extract the tag name from the sentence, those sentences are as mentioned above "I am interested in <tag>", "I love <tag>" or "I like <tag>" where tag is the name of the topic the user is interested in.

```
private String ExtractTagFromUserInput(string userInput) {
            string tag = "";
            if (userInput.ToLower().Contains("like")) {
                 tag = userInput.After("like");
            }

            else if (userInput.ToLower().Contains("love")) {
                tag = userInput.After("love");
            }

            else if (userInput.ToLower().Contains("interested in")) {
                tag = userInput.After("interested in");
            }

            return tag;
        }
```

Code snippet 7 extracting the tag from the sentence using Contains() and After() methods inside TalkToSeeaScreen.xaml.cs class in portable project

Generating SEEA response

This chapter will discuss the process of generating the response to the users input either via voice or text. All the responses generated rules can be found in the Chatbot class, however this documentation focuses on the responses related to the Tags System.

- ***Generating the textual response***

SEEA generates the response by combining both the extracted tag name form the users input, and the pre-entered response string located inside the resource translation resx file named “AppResources.resx”.

```
<data name="seeaResponse" xml:space="preserve">
<value>has been saved and you will recieve suggestions based on that.
       What else are you interested in ?</value>
</data>
```

Code snippet 8 SEEA response upon successfully saving the users entered tag name

The combination process is located inside the TalkToSeeaScreen.xaml.cs file. The combination process does simply concatenate the name of the tag extracted and the loaded translated response from the resource file.

```
public String CreateSEEAResponse(string tagName)
{
 String seeaResponse = ExtractTagFromUserInput(tagName) +
                       " "+AppResources.seeaResponse;
 return seeaResponse;
}
```

Code snippet 9 CreateSEEAResponse method which handles the combination process

The result of this phase is a string containing the SEEA response textually, this string is being placed inside the frame of SEEA response.

The frame of SEEA response is being created by code, and not using a UI builder, and can be found inside the SeeaTalkFrame.cs file.

This frame contains a Text property that shall contain the SEEA response.

In order to display the frame on the chat screen, it shall be added to the stacklayout called “chat_messages_stack” (see code snippet 1).

chat_messages_stack contains both the users as well as SEEA messages, and what is differentiating those two to the user is their location on the screen, whether on the left or the right side as well as their colors, where SEEA responses are presented with a shade of red and the users are in white.

```
<StackLayout x:Name="chat_messages_stack"
             VerticalOptions="StartAndExpand"/>
```

Code snippet 10 chat_messages_stack declaration in TalkToSeea.xaml

In case the user did not use the virtual keyboard to enter the input but rather the voice, the textual response generated shall be then parsed to a speech response using the Xam.plugins.TextToSpeech 4.0.0.7 plugin.

The usage of this plugin is done via calling the Speak() function inside the CrossTextToSpeech class implemented in the plugin installed.

TriggerSeeaRespond() method checks which input entering method has been used by checking the visible views, either the text or the speech view, to determine whether to call the Speak() method or not after adding the SEEA chat frame to the chat stacklayout as shown in the following code snippet.

```
public void TriggerSeeaRespond(String userInput) {

      //call the method to create the response.
      String seeaRespond = CreateSEEAResponse(userInput);

      //add the response to the screen by calling SEEA side adding method.
      add_new_seea_msg(seeaRespond);

      //speak out loud the response if the user input was via voice.
      if (input_text.IsVisible == false) {
          CrossTextToSpeech.Current.Speak(seeaRespond);
         }
    }
```

Code snippet 11 TriggerSeeaRespond() implemented in TalkToSeeaScreen.xaml.cs

5.6.2 Search for events

Search for events feature consists of three main screens, where in the first screen implemented in “SearchEventScreen.xaml.cs”, the user would enter the search criteria, or leave it empty to view all events available. The event date entered by the user has to follow a set of rules in order for it to be accepted, those rules are as follows.

- The date can not be in more than five years from the date of search.
- The date entered must be in the format of **DD.MM.YYYY.**
- The date must consist of only numbers. No letters are allowed.
- The date must be logically correct, so the day must be in the interval between 1 and 31, the month between 1 and 12 and the year in maximum 5 years from now.

```
private bool IsDateAccepted(String[] dateGiven)
        {
            if (IsSizeOfDate(dateGiven)
                && IsDateJustNumbers(dateGiven)
                && IsDateLogicallyCorrect(dateGiven)
                &&IsDateInTheFuture(dateGiven))
            {
                            return true;
            }
            return false;
        }
```

Code snippet 12 the function that calls the four indicated checks

The second screen is implemented in “SearchResult.xaml.cs” and is responsible of searching for the results using the criteria passed from the previous screen and presenting the search results in a table, or show a message indicating that there were no results found in case the criteria entered did not match any of the available events.

The search for event feature uses the following plugins.

- ***Xam.plugins.Forms.ImageCircle 3.0.0.5***

This plugin has been used to preview the events image in a circle mask inside the resulted events table.

- ***Rg.Plugins.Popup 1.1.5.188***

This plugin is used in particular to view the extra filter options available on the search result screen by clicking on the top right burger menu button.

The third screen is implemented in “EventDetails.xaml.cs” in which the tapped event information on the previous screen is shown to the user, so that the user can decide whether to book the event, add the event to the liked events list, add the event information to an existing calendar on the device or/and view the events location on the official map app offered by the devices operating system manufacturer.

This screen has used a multiple of external plugins in order for it to deliver the extra functionality, those plugins are as follows.

- ***Plugin.Persmissions 3.0.0.12***

This plugin is used to obtain the permissions from the user, when the user taps on the calendar icon in the events details page.

- ***CClarke.Plugin.Calendars 1.0.0***

This plugin is used after the permissions being granted by the user to access the calendars.

The plugin access the calendars and then preview their names to the user, so that the user can choose which calendar to save the events to.

The calendars are shown in a popup view, which has been implemented using the Popup plugin mentioned previously.

5.6.3 Get Inspired

Due to the time restriction and the lack of man resources in the team, this feature had to be dropped in this version.

5.6.4 Profile

Profile screen contains the users saved personal data on the Swiss Engineering platform database. The user is able to preview the name, email, address and a personal photo (see figure 40).

The profile screen has also a button that enables the user to navigate to the screen of the booked events.

The views appeared on the BookedEvents.xaml are decided based on the size of the booked events list. If there are no booked events then a warning label will be shown instead the table of the booked events.

```
public void DecideAppearingViews()
        {
//hide the table view and show the warning label if no events booked were
found
            if (StaticLists.bookedEventsList.Count == 0)
            {
                eventsTable.IsVisible = false;
                emptyListLabelIsVisible.IsVisible = true;
            }

    //if events were found, then show the table and hide the warning label.
            else
            {
                emptyListLabelIsVisible.IsVisible = false;
                eventsTable.IsVisible = true;
            }
        }
```

Code snippet 13 decide which views to appear in BookedEvents.xaml.cs

The booked events are presented in a chronological order, where the events that are on the same date are grouped under the same header, which is the date.

The way those booked events are grouped and ordered in code is by using a LINQ inside the BookedEvents.xaml.cs file.

This LINQ creates a Collection that contains a list of a list of events, in other words, for each index there exist a key which is a date string, and a value which is another list, this list is the events that are in the same date as the header stringfied date.

```
var items = from eventAvaliable in StaticLists.bookedEventsList
            orderby eventAvaliable.EventDate.ToString()
            group eventAvaliable by
            eventAvaliable.EventDate.Substring(0,10)
            into eventsGroup
            select new Grouping<String,Event>(eventsGroup.Key.ToString(),
            eventsGroup);
```

Code snippet 14 LINQ inside BookedEvents.xaml.cs

When the user clicks on an event from the list, the user will navigate to the EventDetails.xaml page, where the tapped events data will be displayed.

5.6.5 Saves Locker

Saves Locker is a tabbed page that contains both the saved events as well as the saved searches (see figure 42 and figure 43).

The user can navigate between those two by tapping on the navigation bar located at the bottom of the screen for both iOS and Android.

The placement of the navigation bar is by default on iOS at the bottom of the screen, but in order to alter this behavior to work for Android as well, the SavesLockerScreen.xaml file description at the top has to be edited to contain two lines of code shown on the following code snippet 15.

```
xmlns:android=
"clr-namespace:Xamarin.Forms.PlatformConfiguration.AndroidSpecific;
assembly=Xamarin.Forms.Core"
android:TabbedPage.ToolbarPlacement="Bottom">
```

Code snippet 15 altering Android navigation bar placement in SavesLockerScreen.xaml

- ***Saved Searches screen***

The saved searches screen as shown in Figure 42 contains a list of the saved searches. Tapping on those searches will initiate a search query with the values indicated on the tapped table cell applied as a search criteria. Those criteria can be accessed and passed as arguments of the tapped item.

```
public async void ToSearchResults(object sender, ItemTappedEventArgs
eventArgs)
        {
   Search eventTapped = eventArgs.Item as Search;
   await Navigation.PushAsync(new SearchResult(eventTapped.SearchText,
                              eventTapped.Date, eventTapped.Place,
                              eventTapped.Organization,
                              eventTapped.EventType));
        }
```

Code snippet 16 initiating a search query using the tapped search arguments

Tapping on the recycle bin located inside the cell will lead to delete the saved search record.

This delete process can be done by binding the image of the recycle bin inside the cell to the content “search criteria” of the cell itself.

Therefor when a recycle bin is being tapped on, a command will be executed to remove the search object binded to that recycle bin from the list of saved searches.

```
private void DeleteSearch(object sender, EventArgs e)
        {

            var image = sender as Image;
            var searchToRemove = image?.BindingContext as Search;

      //execute the command only on the cell that its image was tapped on.
            RemoveSearchCommand.Execute(searchToRemove);

            //refresh the content of the table.
            decideAppearingViews();
        }

        public Command<Search> RemoveSearchCommand
        {
            get
            {
                return new Command<Search>(
                    (Search search) => {
                        StaticLists.savedSearches.Remove(search);
                    });
            }
        }
```

Code snippet 17 removing a search object binded to a recycle bin image

If the list of saved searches is empty, then a label shall be displayed to indicate that no search criteria has been saved yet.

```
public void decideAppearingViews()
        {
            if (StaticLists.savedSearches.Count == 0)
            {
                searchesTable.IsVisible = false;
                emptyListLabelIsVisible.IsVisible = true;
            }

            else
            {
                emptyListLabelIsVisible.IsVisible = false;
                searchesTable.IsVisible = true;
            }
        }
```

Code snippet 18 DecideAppearingViews() controls what views to show according to the number of elements inside the booked events list

- ***Saved Events screen***

The saved events screen is the other tabbed page inside the Saves Locker.

This screen contains all the events that the user liked or saved by clicking on the heart image bottom right (see figure 39) in the events detail screen.

The list of saved events (see figure 43) may contain either a table showing a list of saved events by the user, or a warning label indicating that there were no saved events found.

The method OnAppearing() [53], which is a virtual overridden method that is declared in the Xamarin.Forms namespace, decides which of both views (either the warning label that no saved events exist, or the table of saved events) to show, based on the number of elements inside the saved events list.

```
protected override void OnAppearing()
        {
   likedEventsList = CreateListOfLikedEvents(StaticLists.eventList)
                     .ToList<Event>();

   //if no liked events were found, then show the warning label and hide
the table.
            if (likedEventsList.Count == 0)
            {
                emptyListLabelIsVisible.IsVisible = true;
                eventsTable.IsVisible = false;
            }

    //otherwise show the table of liked events and hide the warning label.
            else if (likedEventsList.Count != 0)
            {

                eventsTable.IsVisible = true;
                emptyListLabelIsVisible.IsVisible = false;
            }
            eventsTable.ItemsSource = likedEventsList;
        }
```

Code snippet 19 OnAppearing() inside SavedEvents.xaml.cs

The saved events are not stored inside a separate list, rather is IsSaved a property of Event object, so when the user saves an event, this property is set to true, and by default it is set to false.

This helps to toggle either the full heart figure or the empty heart in the Event Details screen, by checking if the IsSaved property of that specific event displayed, whether it is set to be true or false, where true will be presented then as the full heart, and false by the empty heart.

Inside the SavedEvents.xaml.cs a new temporary list is created to contain only the saved events, in order to present those on the table.

```
public static List<Event> CreateListOfLikedEvents(IList<Event>
listOfAllEvents)
        {
            List<Event> matchedList = new List<Event>();
             foreach (Event e in listOfAllEvents)
            {
                if (e.IsSaved == true)
                {
                    matchedList.Add(e);
                }
            }
            return matchedList.ToList();
        }
```

Code snippet 20 CreateListOfLikedEvents() iterates through the list of all events to check whether they have been saved or not

When the user clicks on an event displayed, the user will be navigated to the event details screen, where the user can book the event, or un-save it.

After un-saving the event, the event will not be shown anymore inside the list of saved events when the user clicks on the back arrow inside the event details page because the OnAppearing() method mentioned earlier updates the list each time the screen of saved events is displayed.

6 Analysis and future thoughts

This chapter will discuss the final thoughts, self-review for the application presented as well as future improvements that shall be considered in the up-coming versions.

- ***Conclusion***

Understanding what elements does XAMARIN offer was essential for the developers to know what it offers and know its limitations. The simplest type of designs was necessary, so that the persona can deal with the application and have no problems understanding how to interact with the user interface elements and navigate through the available screens. Implementing a simple design is described as one of the best practices of the user interface design. [54]

The main screen includes all the core features buttons, distributed into two groups, while the main feature of the app, which is the voice assistant feature, is represented at the top in full width.

The voice assistant feature is implemented to work only with the English language, a further work is required to enable this feature in the other languages.

The voice feature was the core of this project, and the overall design of the application functionality was designed around it.

Although the initial idea was to use one of the well-known Voice Assistants (e.g Google Assistant or Amazon Alexa), but the scope and the functionality designed for this feature made it not possible to use either of those two. The reason is that Google Assistant designed actions can only work within the scope of the Google Assistant, and not inside another third party application (at least for the time this report was written).

- ***What went well***

Most of the features presented were covered by this show-case application, although not in the same way they were thought to be at the beginning of the semester.

The solution ideas are meant to be capable of overcoming all the obstacles that the website has, and the added value was present in most of the ideas thought of and well documented.

The plugins used were well integrated to achieve their purpose of existence.

The voice feature shaped the overall design of the application, and the brainstormed ideas for the voice assistant enabled the team to add more functionality to the application which lead to extra values for the end users (e.g Talk To SEEA and Get Inspired).

The voice assistant works despite it is only functioning in English. Using the built in on device Voice To Text API saved the time of integrating an external plugin and then maintain its functionality and integration within the app in the future. Additionally, it provides an acceptable level of sentences recognition, as the operating system's company is responsible of enhancing its accuracy and complex terminologies level of recognition.

- ***Lessons learned***

A sufficient time should have been planned for the knowledge-acquisition or self-learning at the beginning to know the capabilities of Xamarin better and learn C# programming as well, because of that some functions, that were implemented at the beginning, could have been done in a less resources demanding approach.

Due to the work-load, the time reserved to learn C# and Xamarin were spent either working on the documentation, designs, or implementing the solution already.

Although the knowledge level increased within the time, but there was no sufficient time at the end to refine all what has been implemented in early stages.

The project result is a technical prototype with a mocked database of events. It is not connected to the API of Swiss Engineering platform database due to variation between the data required by this prototype and the database structure of the Swiss Engineering database.

The lack of consistency on the Swiss Engineering database structure made it even harder to connect to their database (e.g the description of events on the Swiss Engineering database contains the date, location, price … etc).

This prototype is not capable of booking events, as the Swiss Engineering platform does not offer a consistent approach to book events (i.e booking is for some events doable by clicking a button, by others a direct contact with the organizers is required).

The simple technique of extracting the tag from the end-user's input in the "Talk To SEEA" feature by using regular expressions works only for the pre-defined trigger sentences, while some times the Voice To Text built-in API makes mistakes, the application should have a way to delete the mis-understood unintended tags.

The alignment of some of the user interface elements is not perfect, which may cause that some elements overlap with each other.

The second layer of filtering the search event results does not work, and only there as a placeholder. This should have work if more time could be spent re-structuring the objects and their properties, and also learning how to implement the filtering where two screens are involved "one containing the results and the other contains the filtering options".

This show-case application does not reflect exactly the design provided by the team, the different screen sizes that this prototype may be deployed on made it harder to settle on one layout ratio, as Android devices have different screen sizes, the same applies on iOS devices.

- ***Future improvements***

The requirements gathering process didn't include the end-user from the Swiss Engineering Organization, but a sample of potential end-users who are indeed engineers from different fields and backgrounds were consulted in order to know what do they really need in an event app, and what information do they count on to make a booking decision. Nevertheless for a future update, the Swiss Engineering end-users should be consulted and be involved to refine the design and data offered.

The implemented voice assistant needs to be enhanced in the future to handle different context. As for now it can only identify and respond to three phrases exclusively.

A major limitation that was faced during the implementation and designing phase, is the lack of an appropriate data base structure to include all the new attributes that must be saved for each end-user, for example the liked tags and a well-defined attributes information, as the content of the location attribute was not only the actual address, but an inconsistent address information such as only the name of the location, and sometimes the street name and number without postal code. So a re-built database should make the output and displayed information more reliable on the app.

The Swiss Engineering database needs a complete re-design to include all the new attributes used in this show-case app, such as the tags for an event, the description in multiple languages in order for the details to be consistent with the languages that the app supports and also the user personal picture, which is presented in the profile screen.

If the database could not be re-structured, then the next version of this prototype should be capable of extracting the required data from the database attributes available on the current database in use.

A systematic tracking of user behavior and event participation will allow tracking and recommending events. The recommendation is based on common interests with other users which is obtained by attending same events in the past.

This tracking can be even enhanced in the future by establishing a bi-directional recommendation line between two members, when a specified number of liked tags is shared between both members (e.g if the threshold is set to 3, then if both members liked Tag A, Tag B, and Tag C, then future bookings of the first member should be suggested to the second member and vice versa).

Finding a suitable assistant service to accomplish the tasks requested, and as the options are limited from the start to choose from three candidates, which are the major player nowadays on the market, namely Google Assistant, Apple Siri and Amazon Alexa, we advise to go with the Google Assistant service for a number of reasons, the most important may be the ease of use, the huge infrastructure, and the tutorials available which are offered by Google.

In combination with Google Assistant, choosing DialogFlow might be the best option as it was recently acquired recently by Google, therefor we expect this chatbot platform to be supported more by Google and have extra features available to work along Google Assistant.

An extended list of features should be also added to the voice assistant deployed to expand the services it offers, and not only to save tags and view answer simple questions, but to book events directly for instance.

The process of reserving events is too inconsistent throughout the different Swiss Engineering chapters and even varies from one event to another. Sometimes the user signs up by contacting the organizers, other times direct booking can occur through the website.

A consistent process of booking can benefit the user experience for the entire Swiss Engineering Association in the future.

7 Refrences

[1] “What is User Centered Design?,” *The Interaction Design Foundation*. [Online]. Available: https://www.interaction-design.org/literature/topics/user-centered-design. [Accessed: 16-Jul-2019].

[2] S. Gladkiy, “User-Centered Design: Process and Benefits,” *UX Planet*, 14-Jun-2018. [Online]. Available: https://uxplanet.org/user-centered-design-process-and-benefits-fd9e431eb5a9. [Accessed: 12-Jul-2019].

[3] J. Chen, “Neural Network Definition,” *Investopedia*. [Online]. Available: https://www.investopedia.com/terms/n/neuralnetwork.asp. [Accessed: 31-Jul-2019].

[4] A. Virdee, “Data Augmentation Experimentation,” *Medium*, 09-Jun-2018. [Online]. Available: https://towardsdatascience.com/data-augmentation-experimentation-3e274504f04b. [Accessed: 31-Jul-2019].

[5] “SpecAugment: A New Data Augmentation Method for Automatic Speech Recognition,” *Google AI Blog*. .

[6] S. Toshniwal *et al.*, “Multilingual Speech Recognition With A Single End-To-End Model,” *ArXiv171101694 Cs Eess*, Nov. 2017.

[7] S. Petridis, J. Shen, D. Cetin, and M. Pantic, “Visual-Only Recognition of Normal, Whispered and Silent Speech,” *ArXiv180206399 Cs*, Feb. 2018.

[8] SpeechAngel, “The difference between speaker-dependent and speaker-independent recognition software,” *SpeechAngel*, 04-May-2016. .

[9] R. Lopez-Ruiz, *From Natural to Artificial Intelligence: Algorithms and Applications*. BoD – Books on Demand, 2018.

[10] K. I. Nordstrom and P. F. Driessen, “Variable Pre-Emphasis LPC for Modeling Vocal Effort in the Singing Voice,” p. 4, 2006.

[11] N. Dave, “Feature Extraction Methods LPC , PLP and MFCC In Speech Recognition.”

[12] H. Hermansky and N. Morgan, “RASTA processing of speech,” *IEEE Trans Speech Audio Process.*, vol. 2, pp. 578–589, 1994.

[13] N. S. Uchat, “Hidden Markov Model and Speech Recognition,” p. 26.

[14] P. P. Kajave, “Microsoft Word - Finding More Non-supersingular Elliptic Curves for Pairing..,” 2009.

[15] A. H. Mansour, G. Z. A. Salh, and K. A. Mohammed, “Voice Recognition using Dynamic Time Warping and Mel-Frequency Cepstral Coefficients Algorithms,” 2015.

[16] T. Kinnunen, “Comparison of clustering algorithms in speaker identification,” 2000.

[17] D. H. B. Kekre and M. Kulkarni, “Speaker Identification by using Vector Quantization,” 2010.

[18] “Understanding the impact of speech recognition software on search,” *Search Engine Watch*, 29-Sep-2016. .

[19] “Leveraging speech recognition technology in call centers,” *SearchCustomerExperience*, May-2009. [Online]. Available: https://searchcustomerexperience.techtarget.com/report/Leveraging-speech-recognition-technology-in-call-centers. [Accessed: 09-Jul-2019].

[20] “Voice and Speech Recognition,” *FindBiometrics*. [Online]. Available: https://findbiometrics.com/solutions/voice-speech-recognition/. [Accessed: 09-Jul-2019].

[21] “How Voice Assistants Are Changing Our Lives,” *Smartsheet*, 16-Apr-2018. [Online]. Available: https://www.smartsheet.com/voice-assistants-artificial-intelligence. [Accessed: 09-Jul-2019].

[22] J.-0552-M. Campbell, “Speaker Recognition for Forensic Applications,” p. 34, 2014.

[23] “Google Assistant | Dialogflow.” [Online]. Available: https://dialogflow.com/docs/integrations/google-assistant. [Accessed: 11-Jul-2019].

[24] “Entities overview | Dialogflow.” [Online]. Available: https://dialogflow.com/docs/entities. [Accessed: 11-Jul-2019].

[25] “Intents | Actions on Google,” *Google Developers*. [Online]. Available: https://developers.google.com/actions/reference/rest/intents. [Accessed: 11-Jul-2019].

[26] “What is Google Assistant and what can it do? - Pocket-lint.” [Online]. Available: https://www.pocket-lint.com/apps/news/google/137722-what-is-google-assistant-how-does-it-work-and-which-devices-offer-it. [Accessed: 05-Aug-2019].

[27] “SiriKit | Apple Developer Documentation.” [Online]. Available: https://developer.apple.com/documentation/sirikit. [Accessed: 11-Jul-2019].

[28] “Manage an HTTP/2 Connection with AVS | Alexa Voice Service.” [Online]. Available: https://developer.amazon.com/docs/alexa-voice-service/manage-http2-connection.html. [Accessed: 11-Jul-2019].

[29] “Interaction Model | Alexa Voice Service.” [Online]. Available: https://developer.amazon.com/docs/alexa-voice-service/interaction-model.html. [Accessed: 12-Jul-2019].

[30] “Capabilities API | Alexa Voice Service.” [Online]. Available: https://developer.amazon.com/docs/alexa-voice-service/capabilities-api.html#use-cases. [Accessed: 11-Jul-2019].

[31] “Xam.Plugins.TextToSpeech 4.0.0.7.” [Online]. Available: https://www.nuget.org/packages/Xam.Plugins.TextToSpeech/. [Accessed: 16-Jul-2019].

[32] “Google.Cloud.TextToSpeech.V1 1.0.0.” [Online]. Available: https://www.nuget.org/packages/Google.Cloud.TextToSpeech.V1/. [Accessed: 16-Jul-2019].

[33] “Supported platforms | Google Cloud APIs.” [Online]. Available: https://googleapis.github.io/google-cloud-dotnet/docs/guides/platforms.html. [Accessed: 16-Jul-2019].

[34] “Snow.Xam.Plugins.TextToSpeech 2.0.2.” [Online]. Available: https://www.nuget.org/packages/Snow.Xam.Plugins.TextToSpeech/. [Accessed: 16-Jul-2019].

[35] “Top 10 Cross-Platform Mobile Development Tools,” *Hongkiat*, 17-Aug-2016. [Online]. Available: https://www.hongkiat.com/blog/cross-mobile-platform-framework-wora/. [Accessed: 11-Jul-2019].

[36] “Why developers like Gluon,” *StackShare*. [Online]. Available: https://stackshare.io/gluon. [Accessed: 11-Jul-2019].

[37] “What is Flutter? Benefits and limitations,” *Codemagic blog*, 18-Jan-2019. [Online]. Available: https://blog.codemagic.io/what-is-flutter-benefits-and-limitations/. [Accessed: 05-Aug-2019].

[38] “New Features & Improvement in Google Flutter 1.2,” *Angular Minds*, 25-Apr-2019. [Online]. Available: https://www.angularminds.com/blog/article/google-flutter-new-features-and-improvements.html. [Accessed: 05-Aug-2019].

[39] “Flutter: Pros and Cons for Seamless Cross Platform Development.” [Online]. Available: https://hackernoon.com/flutter-pros-and-cons-for-seamless-cross-platform-development-c81bde5a4083. [Accessed: 05-Aug-2019].

[40] “The Pros and Cons of Progressive Web Apps | JABA Blogs Adelaide.” [Online]. Available: https://jaba.com.au/blog/pros-and-cons-of-progressive-web-apps. [Accessed: 05-Aug-2019].

[41] “Dialogflow vs Lex vs Watson vs Wit vs Azure Bot | What to Choose?,” *Kommunicate Blog*, 10-May-2019. .

[42] “Languages | Dialogflow Documentation,” *Google Cloud*. [Online]. Available: https://cloud.google.com/dialogflow/docs/reference/language. [Accessed: 11-Jul-2019].

[43] “Limits - Amazon Lex.” [Online]. Available: https://docs.aws.amazon.com/lex/latest/dg/gl-limits.html. [Accessed: 18-Jul-2019].

[44] “Watson Assistant Pricing | IBM Cloud.” [Online]. Available: https://www.ibm.com/cloud/watson-assistant/pricing/. [Accessed: 11-Jul-2019].

[45] “Supported languages.” [Online]. Available: https://cloud.ibm.com/docs/services/assistant?topic=assistant-language-support#language-support. [Accessed: 11-Jul-2019].

[46] “Wit.ai.” [Online]. Available: https://wit.ai/faq%EF%BB%BF. [Accessed: 11-Jul-2019].

[47] “Why Great Designers Steal—and Are Proud of It :: UXmatters.” [Online]. Available: https://www.uxmatters.com/mt/archives/2011/04/why-great-designers-stealand-are-proud-of-it.php. [Accessed: 18-Jul-2019].

[48] maddyleger1, “XAML Previewer for Xamarin.Forms - Xamarin.” [Online]. Available: https://docs.microsoft.com/en-us/xamarin/xamarin-forms/xaml/xaml-previewer/. [Accessed: 08-Aug-2019].

[49] “Xam.Plugins.Forms.ImageCircle 3.0.0.5.” [Online]. Available: https://www.nuget.org/packages/Xam.Plugins.Forms.ImageCircle/. [Accessed: 20-Jul-2019].

[50] “Rg.Plugins.Popup 1.1.5.188.” [Online]. Available: https://www.nuget.org/packages/Rg.Plugins.Popup/. [Accessed: 20-Jul-2019].

[51] “GitHub - jamesmontemagno/PermissionsPlugin: Check and Request Permissions Plugin for Xamarin and Windows.” [Online]. Available: https://github.com/jamesmontemagno/PermissionsPlugin. [Accessed: 20-Jul-2019].

[52] C. Clarke, *Cross-platform calendar API plugin for Xamarin and Windows: TheAlmightyBob/Calendars*. 2019.

[53] dotnet-bot, “Page.OnAppearing Method (Xamarin.Forms).” [Online]. Available: https://docs.microsoft.com/en-us/dotnet/api/xamarin.forms.page.onappearing. [Accessed: 24-Jul-2019].

[54] “User Interface Design Basics,” 21-May-2014. [Online]. Available: /what-and-why/user-interface-design.html. [Accessed: 09-Aug-2019].

8 Glossary

Term	Description
User Experience	The overall experience of a person using a product such as a website or computer application, especially in terms of how easy or pleasing it is to use.
User Interface	The means by which the user and a computer system interact, in particular the use of input devices and software.
Artificial Intelligence	the theory and development of computer systems able to perform tasks normally requiring human intelligence, such as visual perception, speech recognition, decision-making, and translation between languages.
API **"Application Programming Interface "**	a set of functions and procedures allowing the creation of applications that access the features or data of an operating system, application, or other service.
Mobile Application	Is a type of application software designed to run on a mobile device, such as a smartphone or tablet computer.
Prototype	A first or preliminary version of a device or a software from which other forms are developed.
Mock-up	A model or replica of a machine or structure, used for instructional or experimental purposes.
Voice Recognition	Computer analysis of the human voice, especially for the purposes of interpreting words and phrases or identifying an individual voice.
Chatbot	A computer program designed to simulate conversation with human users, especially over the Internet.

Rule-based Chatbot	Rule-based chatbots are also referred to as decision-tree bots. As the name suggests, they are powered through a series of defined rules. These rules are the basis for the types of problems the chatbot is familiar with and can deliver solutions for.
Artificial Intelligence Chatbot	AI chatbots that use machine learning are built to understand the context and intent of a question before formulating a response.
Cross-Platform Application development	Is the process of creating mobile apps that can be deployed or published on multiple platforms using a single codebase, instead of having to develop the app multiple times using the respective native technologies for each platform.
Plugin	Is a software component that adds a specific feature to an existing computer program. When a program supports plug-ins, it enables customization.

9 Appendix

This chapter includes all of the main materials that was produced during the making of this project.

9.1 Vision

The following is the vision that was created at the beginning of this projects, and was updated through the life cycle of this project implementation.

VISION

What is your purpose for creating the product?

Which positive change should it bring about?

Book your event at a glance!

- The “Swiss Engineering Event App” is here to book events on user’s smartphone instead of using the website, as we realized that the website’s mobile usage is not as smooth and intuitive as it is intended to be.
- We want to build an app with a simple modern design that is not overloaded with too many features that the user does not use on a regular basis or even at all.
- “Make It Simple To Use” is the core objective that we aim for while building this app, using the UX guidelines while also keeping in mind that there will be no tutorial are the fundamentals and the base that “Swiss Engineering Event App” will be built on.
- The end-user does not have time to spare; that is why the “Swiss Engineering Event App” will require less steps than the website to book events and edit event’s reservations.
- The core functions of the “Swiss Engineering Event App” should be visible at the top and easier to access than the extra features, which the user can still access using a logical sequence based on the user’s previous experiences of dealing with smartphones apps.
- The “Swiss Engineering Event App” will have the Voice Assistant feature included in a way that facilitate exploring the events landscape, details as well as booking events.

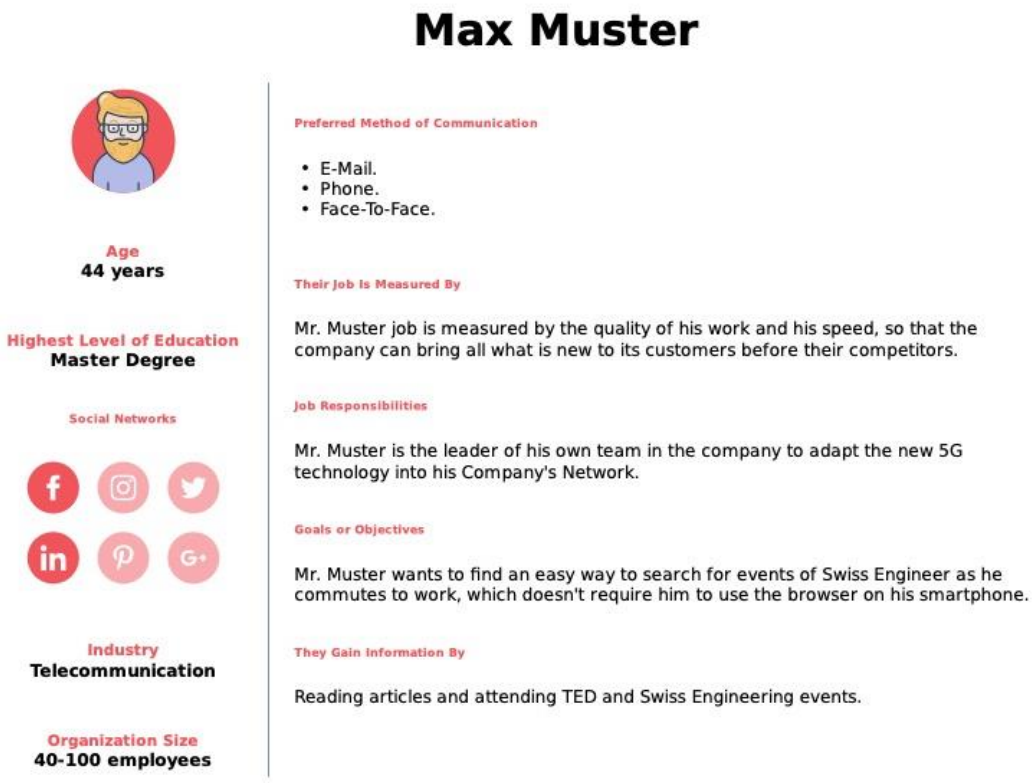
Target Group *Which market or market segment does the product address?* *who are the target customers and users?*	**Needs** *What problem does the product solve?* *which benefit does it provide?*	**Product** *What product is it?* *what makes it stand out?* *is it feasible to develop the product?*	**Business goals** *How is the product going to benefit the company?* *what are the business goals?*
• The Market Segment that the app targets, are the “Swiss Engineering” members. • The Target users are the members (mostly engineers) who want to book their events. • The registered students are also considered to be an important targeted users as they are future engineers and potential members who will be paying	• Using the website on a relatively smaller screen size raise the risk of unintentionally messing up the layout, especially with a huge number of tabs and navigation menus, such as the number contained in the website; thus the app is needed. • Less elements per screen on the app, compared to a laptop, will keep the user fully focused and never distracted. • The app will always be easier and more	• The “Swiss Engineering Event App” is a smartphone cross-platform app that runs on both iOS and Android natively. • The “Swiss Engineering Event App” is the only app developed to work with the “Swiss Engineering” platform. As there are no competitors, we believe that the success of the app will depend solely on the member’s willingness to make the switch from using their laptops to reserve their events to using their smartphones instead. We are confident that they will never look back to use	• Because of the impressive app, new members will join the association which would potentially increase the amount of reservations, and therefore the revenue from the hosted events. • The reputation of the association will improve as the number of event visitors increases.

the membership fees in the near future; so the app is designed to retain them.	intuitive to use than a website, especially on a smartphone.	their laptops for reservations ever again, due to the easiness of use and the better experience that the app provides.	

9.2 Stakeholder analysis

The following are the personas that were created for both types of the application's end-users.

The Onion model analysis for the stakeholder's analysis is attached.



Appendix Figure 1 Member end-user

Maccy Mendoza

Age
34 years

Highest Level of Education
Bachelor Degree

Social Networks

Industry
Architecture

Organization Size
20-40 Employee

Preferred Method of Communication

- Face-To-Face.
- Phone.

Their Job Is Measured By

Finishing the desired building blueprint in time so that the constructor can start working.

Job Responsibilities

Ms. Mendoza is a team member and responsible of achieving her her tasks (the building blueprints) which are assigned to her by the team leader.

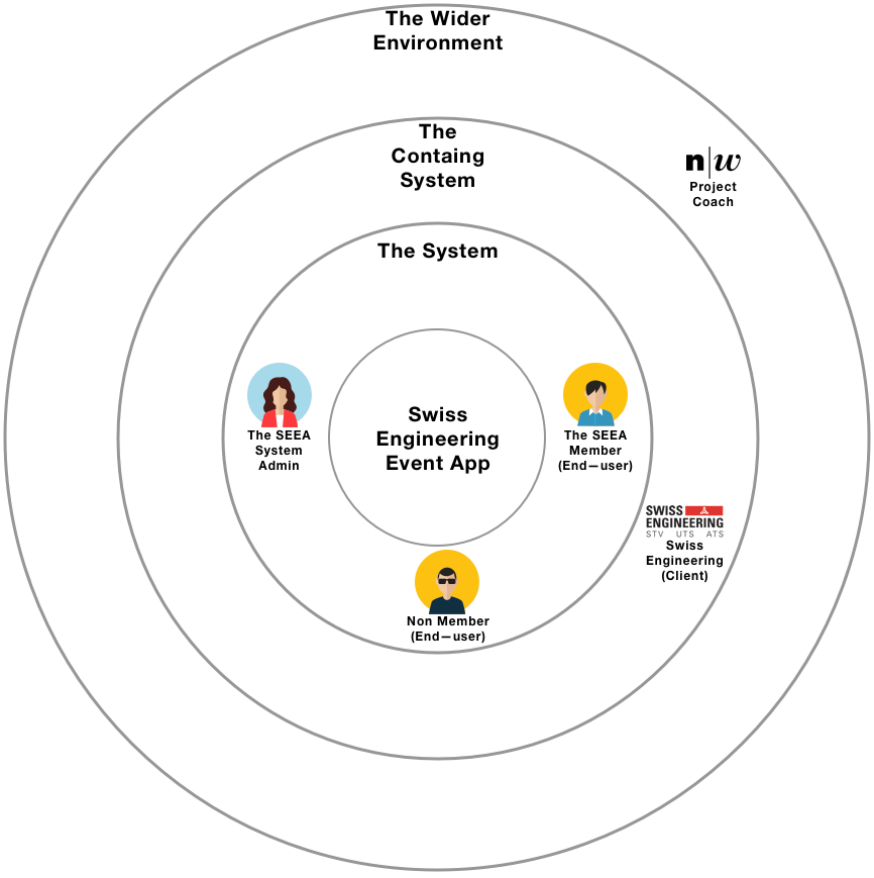
Goals or Objectives

Ms. Mendoza would love to have an app that suggest her events which can help her to know more about the modern building design techniques.

Reports to

Ms. Mendoza has to report to her team leader.

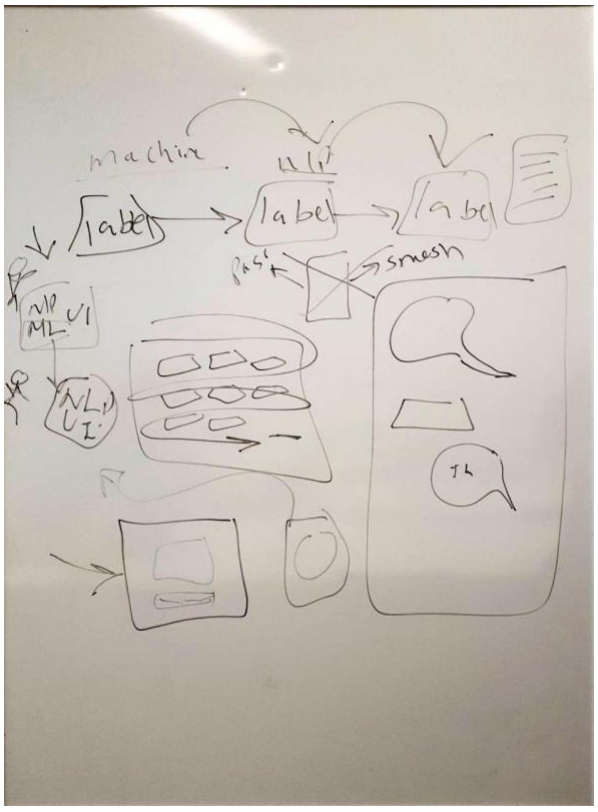
Appendix Figure 2 non-member end-user

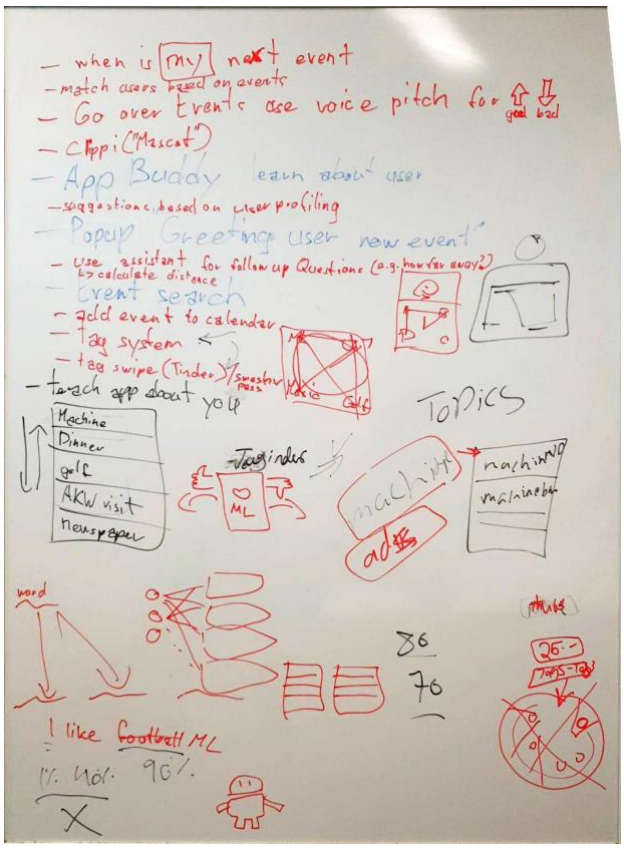


Appendix Figure 3 Stakeholder Onion model analysis

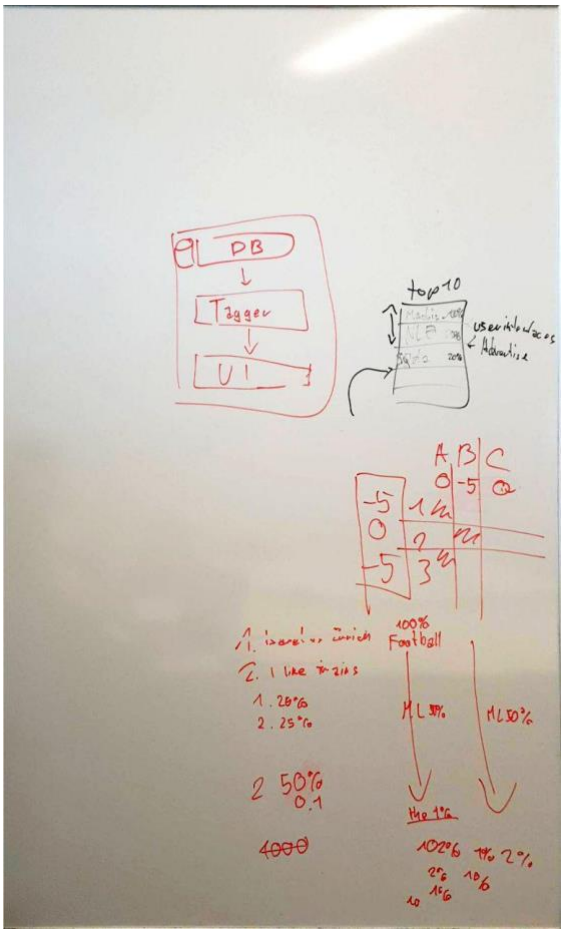
9.3 Brainstorming Boards

The following graphs were made during the features ideas brainstorming process, which lead to the creation of both Get Inspired(Social Factor) and Talk To SEEA features, as well as the explorative features.





Appendix Figure 4 Brainstoming Boards



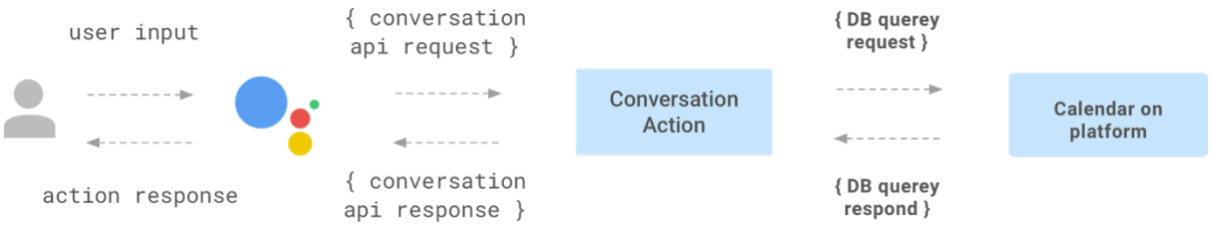
9.4 Brainstorming textual description

This sub-chapter describes the brainstormed ideas from the brainstorming session in text.

When is my next event

This feature helps the end-user to get information about the next booked event. If the end-user has booked **"Does culture matter on international business ?"** event on 4th April, The end-user would say phrases like "Hey Seea, when is my next event ?" and the voice assistant should respond "your next event **"Does culture matter on international business ?"** is tomorrow at 5 PM".

This requires that the voice assistant in use is aware of the end-user booked events.



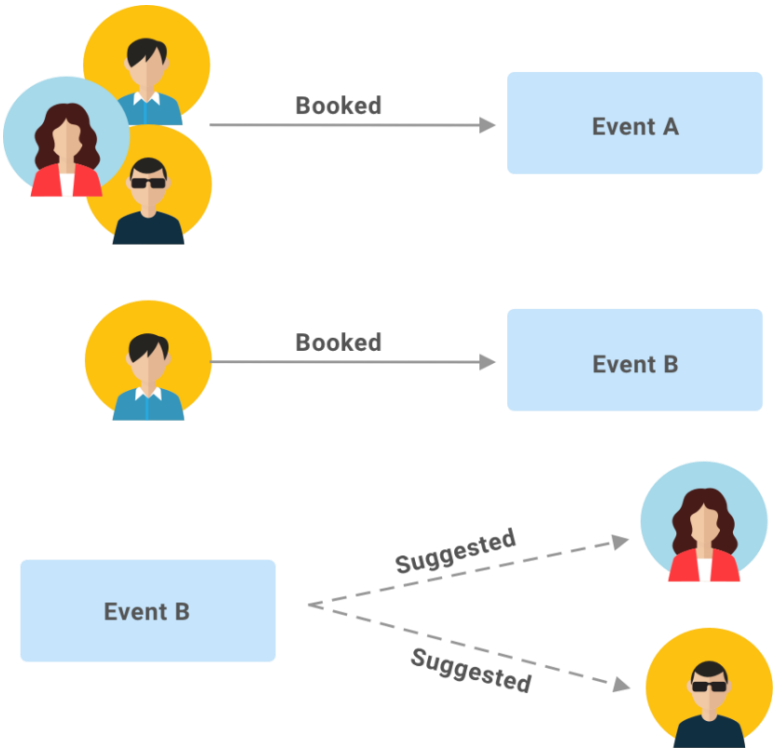
Appendix Figure 5 communicating with DialogFlow using Google Assistant Action

Suggest events based on users of the same taste

When a user books an event, the app would start creating a matching process in the background to suggest new apps to attend, based on what did the co-attendees who visited the same event booked later.

The matching process would have to completely anonymous so that the privacy of the end-user doesn't get violated.

The developer team really liked this idea because it helps to create a social relationships between the end-users who have the same taste and interests. This idea's human value is really high to be just dropped, because it would help to create a real social community around the Swiss Engineering platform.



Appendix Figure 6 Social Factor

Let Seea get to know you

Seea is the name of the voice assistant inside the app, the name is the initials of the project "**S**wiss **E**ngineering **E**vent **A**pp ".

Seea would start asking the end-user several questions to get to know him/her better, and start filling out the gaps to construct a model and start looking what events should it suggest to the end-user.

Those suggestion can be shown whether in the same page after answering all the question, or later in the suggestions screen.

This idea requires that Seea is aware of the events Database and can convert the textual recognized speech into a query to retrieve the results.

Appendix Figure 7 Talk To SEEA

Use Voice Assistant for follow-up questions

We as a developing team understands the different between the human and machine, for instance the human would remember the name of the streets that was raised in, visited as a teenager or had business in as an adult.

One of the first questions that the end-user would after showing an interest in an event, is where is it? even if the end-user saw the name of the street, the user would most probably search for that location to know how far away is it, especially if that event is taking place in another canton.

So this feature would save the time opening the maps and entering the name of the street and choosing the end-user location, as the end-user would ask simple direct questions, such as “How far away is it?” and Seea would reply “10 km” for instance so that the end-user can decide whether or not to book this event.

This feature requires that Seea is aware of the users current location – or the address listed as the address in the user’s profile – and the location of the event.

Event search

The obvious go to feature when voice assistant is to implement in an event app. The end-user would trigger the voice assistant “Seea” and ask her to search for events that match the entered vis speech criteria given.

Seea would then respond with showing the events that match those criteria as a list in the events search results screen.

Entering the criteria could be done in the following two different ways:

- **Input as one sentence**
  The end-user would have to say full sentences, for instance “Show me the events about **economy** on **4th April** in **Zurich**”, then Seea will retrieve the result of searching for those three inputs: Topic, date and location.

- **Input as a conversation**
  Seea would first ask the user about which type of search criteria input to enter via voice, for instance Seea would say “What topics are you interested in?” the end-user would say “Economy”, then Seea would ask about the date by saying “which date should I search for” then end-user would then say “4th April” and Seea would finally ask about the location by saying “in which canton should I search” and the end-user would say “Zurich”.

Of course the feasibility of implementing such a feature is still discussable, because of how intuitive is it to replace the normal text input which most of the potential end-users prefer with a speech input.

Seea has to be connected to the Database of the available events and their corresponding attributes needed to complete this search query.

Add events to calendar

This is one of the most intuitive to use features on this list. Normally when the end-user books an event, the next step is to copy this event information manually in the calendar, or wait for an event invitation via mail to import the event object to the calendar or if the user was lucky enough, the app would offer the end-user a button called “import to calendar” to import the event into the user’s private calendar service.

Seea has a potential usage at this point to import the booked event to the end-user’s calendar by just saying “import this event to my calendar” and that is it, the event is now saved into the user’s default calendar in use.

In this case, Seea has to be aware which calendar is the end-user using and also to know how to construct an event calendar object to import it into the calendar.

TagSquare

This is a gamic feature that might make the process of entering the end-user's preferences more fun than usual. In the screen a square will be shown with a label in the middle containing a name of one of the tags entered as an event was created, the users task is to swipe the tag label into one of the four corners of the square, namely "**book me**, **interesting**, **meh** and **hell no**" each time the user swipes one tag into one of those four corners and release a new tag appear in the middle to repeat the operation again. The end goal is to teach the AI what topics are the end-user interested in to send the end-user better suggestions in the future.

All what is needed is an access to the list of tags used for events, and then a way to save the end-user's choices and use them in the future to help provide even better suggestions.

Appendix Figure 8 Tags square

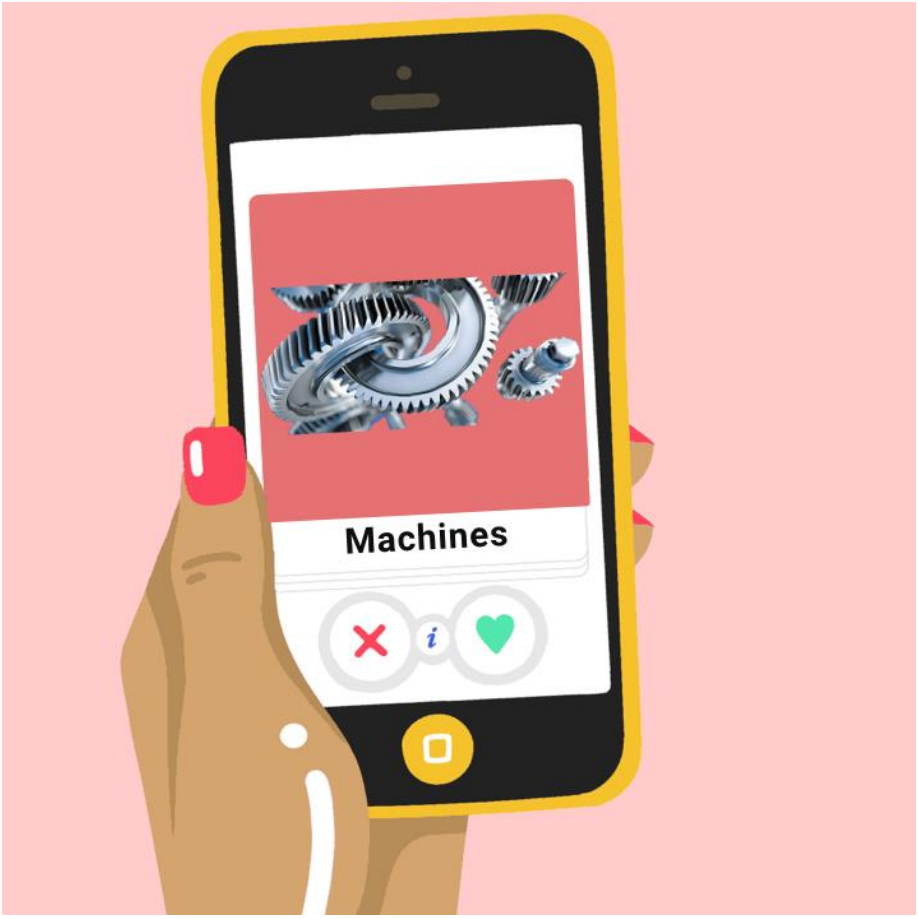
EvenTinder

Dating is back on the table for all involved in a relationship ☺

This idea is basically about using the event's tags to create a list of cards stacked on each other, all what the end-user needs to do is to swipe the card right or left, smash or pass.

The "smashed" cards tags will be used in the future to provide better recommendations in the future for the end-user.

The AI needs to be aware of the tags entered in the database to provide them as cards.



Appendix Figure 9 EventTinder

Seea Mascot

This is another attempt of gamification. The Idea would be to create a visual personal assistant (e.g. Microsoft clippy) that is always present on the screen and would double as an integrated help function. For example, the mascot would notify the user if there is no internet connection present or would be clickable on pages that might require help. The mascot should be located on one of the edges or corners of the screen to be easily accessible without obscuring the view on the actual application and its functions.

The Mascot would need to be aware about the content of each screen so that it give helpful instructions, the team needs to design the Mascot as well.

9.5 Initial Use-cases draft

This sub-chapter previews the initial version of the user-cases that was created in the early stages of this application development.

ID	1.1
Name	Login
Actors	Member End User
Goal	Login into the platform
Pre-Conditions	The User has a membership
Steps	1. The user opens the app. 2. The user taps on the profile icon top left. 3. The user enters the username and password. 4. The user taps on login button.
Alternative	---
Post-Conditions	The user is now logged in.

ID	1.2
Name	Register
Actors	Non Member end-user
Goal	Create an account to be able to reserve events.
Pre-Conditions	The user doesn't have an account.
Steps	1. The user opens the app. 2. The user taps on the profile icon top left. 3. The user taps on register now label.
Alternative	---
Post-Conditions	The user will be redirected to the Swiss Engineering website to complete the process of creating an account.

ID	1.3
Name	Reset Member's Password
Actors	Member end-user
Goal	Reset the Member's forgotten password to be able to login into the member's account.
Pre-Conditions	The user has a valid account.
Steps	1. The user opens the app. 2. The user taps on the profile icon top left. 3. The user taps on "Forgot My Password" label at the bottom.
Alternative	---
Post-Conditions	The user will be redirected to the Swiss Engineering website to complete the process of resetting the account's password.

ID	2.1
Name	Search Events.
Actors	End-User (Whoever has the app installed on a smartphone).
Goal	To explore a list of available events matching the search filters applied.
Pre-Conditions	The app is installed on the smartphone.
Steps	1. The user opens the app. 2. The user taps on events card. 3. The user enters his preferred search criteria to apply as filters. 4. The user taps on search button.
Alternative	---
Post-Conditions	A list of events that matches the entered search criteria will show up in a table.

ID	2.2
Name	Save search filters.
Actors	Member End User
Goal	Save search criteria to be able to use it in the future easily without having to re-enter them again.
Pre-Conditions	The Member has logged in successfully.
Steps	1. The user opens the app. 2. The user navigates to the events card. 3. The user enters the search criteria to be applied while searching. 4. The user taps on the star icon. 5. The user taps on search button.
Alternative	---
Post-Conditions	A list of events that matches the entered search criteria will show up in a table, and the entered criteria will be saved to re-use in the future easily.

ID	2.3
Name	Search using saved search criteria recipe.
Actors	Member End User
Goal	Search quickly using a saved search criteria.
Pre-Conditions	1. The Member has logged in successfully. 2. The Member has a saved search criteria.
Steps	1. The user opens the app. 2. The user taps on the favorite search card. 3. The user taps on one of the search criteria recipe shown.
Alternative	---
Post-Conditions	A list of events that matches the entered search criteria will show up in a table.

ID	2.4
Name	Book event.
Actors	Member End User
Goal	Book a seat in an event.
Pre-Conditions	The Member has logged in successfully.
Steps	1. The user opens the app. 2. The user navigates to the events card. 3. The user enters the search criteria to be applied while searching. 4. The user taps on the star icon. 5. The user taps on search button. 6. The user taps on one of the listed events. 7. The user taps on the reserve button.
Alternative	---
Post-Conditions	The user will be redirected to the Swiss Engineering website to continue the reservation process.

ID	2.5
Name	Delete a search criteria recipe.
Actors	Member End User
Goal	Remove a search criteria saved recipe.
Pre-Conditions	1. The Member has logged in successfully. 2. The Member has a saved search criteria.
Steps	1. The user opens the app. 2. The user taps on the favorite search card. 3. The user swipes the card of the recipe to delete to the left.
Alternative	3b. The user taps on the edit button top left. 4. The user taps on the minus (Negative) symbol beside the recipe to delete.
Post-Conditions	The recipe will not be listed anymore among the saved recipes.

ID	2.6
Name	Read Events Description.
Actors	End User
Goal	View the description of an event with all available information about location, date, topic … etc.
Pre-Conditions	The app must be installed on the smartphone.
Steps	1. The user opens the app. 2. The user taps on events card. 3. The user enters his preferred search criteria to apply as filters. 4. The user taps on search button. 5. The user taps on one of the listed events.
Alternative	---
Post-Conditions	The description of the tapped event will be shown to the user with all included information.

ID	3.1
Name	Cancel reservation.
Actors	Member End User
Goal	Cancel a reservation of an event.
Pre-Conditions	1. The Member has logged in successfully. 2. The Member has a reservation.
Steps	1. The user opens the app. 2. The user taps on “My reservations” card. 3. The user taps on the reservation wished to be canceled. 4. The user taps on cancel button at the bottom of the reservation’s description.
Alternative	---
Post-Conditions	The user will be redirected to the Swiss Engineering website to continue the cancelation process.

ID	3.2
Name	See my reservations.
Actors	Member End User
Goal	Open the list of the user's reserved events.
Pre-Conditions	1. The Member has logged in successfully. 2. The Member has a reservation(s).
Steps	1. The user opens the app. 2. The user taps on "My reservations" card.
Alternative	---
Post-Conditions	A list of the reserved events will be shown in a table.

9.6 Features List

This sub-chapter previews textually the list of the features planned to be included in the Swiss Engineering Event App.

Talk to SEEA

As the technology nowadays is pushing towards the development in the artificial intelligence **AI** field, the client requested to have a smart virtual assistant included in the app, which the user may use to achieve small tasks, with no precise idea given on how to implement it and how should the end-user interact with it, which was left for the project team to discover and implement what suits the end-user and their level of technological experience the best, considering the different levels of technological affinities of the end-user and suitable situations where the assistant might become handy and intuitive to use.

So we came up with the idea of **SEEA**, which is the name of the voice assistant inside the app, the name is the initials of the project "**S**wiss **E**ngineering **E**vent **A**pp ".

Seea would start asking the end-user several questions to get to know him/her better, and start filling out the gaps to construct a model and start looking what events should it suggest to the end-user.

Those suggestion can be shown whether in the same page after answering all the question, or later in the Get Inspired screen.

This idea (as illustrated in appendix figure 7) requires that Seea is aware of the events Database and can convert the textual recognized speech into a query to retrieve the results.

Search for Event

Search event is one of the core features of the Swiss Engineering Event App or any other event app in general.

The client asked to implement this feature considering the old Swiss Engineering App platform which was still in use till the date of writing this chapter. Therefor the search criteria in the Swiss Engineering Event App was inspired from the available filtering/searching options on the web platform, namely Search text, date, place, event type and the organization, it is worth mentioning that the event code was deleted as the client requested on the first Milestone meeting.

Get Inspired

Get Inspired is one of the important features of the app because it translates the end-user's input to the virtual assistant into a textual results which helps to make the app usage more personalized.

Right from the early life phases of the project, the project team decided to have this feature and to use it to offer the end-user more realistic suggestions which match to the end-user personal interest.

The Get Inspired feature is implemented in many apps no matter in which genre it belongs, Facebook market place and YouTube for instance.

With this feature the project team expects the booking rate to be higher than before, because with Get Inspired, events will be suggested, even though the end-user might never search for before.

Get Inspired will also include the reason why is this event being suggested, and the user might choose to never receive suggestions from specific tags again.

Profile

In the profile the user will see the entered personal information, such as the photo uploaded, name, email and the address.

The user will also find in the profile the booked events grouped by their dates so that user can have a quick look at what booked events are coming next. The booked events should work even if the user is not connected to the internet, but the user can not update any booked events by deleting them or book new ones in this case.

In the future the dark theme switch should be added in the profile as well, so that the user who likes the dark theme might use it in the app.

Saves Locker

The Saves Locker idea is to act as the end-user's vault, in which all the saved searches and liked events are kept to enable ease of access to those in the future, that is why this feature is presented on the home screen.

Both liked events and saved searches are combined under the Saves Locker feature and screen, but separated at the same time in two different tabbed pages; one for the saved searches and the other is for the liked events.

Using this feature the end-user doesn't have to re-enter the search text again or any other criteria's attribute if the save search switch was turned on by the time the user enters those attributes in the Search Event screen.

The user might also like an event to save it for later, so a decision can be made afterwards whether or not to book an event.

The Personal Social Feature

According to a Harvard Business Review survey, the engineering were found to be the second loneliest profession, where Law came first.

So the project team decided to implement this social aspect feature in the app to help matching the engineers with other engineers who might have a similar interest.

The **AI** waits till people who booked the same event with the end-user Kevin book another event, then its suggests that event to Kevin in the Get Inspired page, so Kevin will have the opportunity to explore new themes and events which he might never think about before, and also make new friends inside the society.

In more details (see appendix figure 6), When a user books an event, the app would start creating a matching process in the background to suggest new apps to attend, based on what did the co-attendees who visited the same event booked later.

The matching process would have to completely anonymous so that the privacy of the end-user doesn't get violated.

The developer team really liked this idea because it helps to create a social relationships between the end-users who have the same taste and interests. This idea's human value is really high to be just dropped, because it would help to create a real social community around the Swiss Engineering platform.

This feature can not be turned off now, but in the feature it should have a toggle switch in the profile page.

9.7 Requirements Analysis

This chapter lists all the requirements obtained at an early stages of the application development process.

The End-user described in the following requirements is a Swiss Engineering Association member, while the user is the non-member, who downloaded the app.

ID	EPIC	Req	User Story	Acceptance Criteria
1.1	Talk to SEEA	The built in assistant must have a built in virtual assistant which saves the user liked tags(topics) to be used later in the Get Inspired page.	As a member, I want to enter my preferred topics and interests, so that I get suggestions based on those interests.	
1.2	Talk to SEEA	The built in assistant must be capable of suggesting/displaying events in the Get Inspired page based on the user's entered interests(tags/topics).	-	
1.3	Talk to SEEA	The built in assistant must be capable of understanding simple questions related to the user's booked events and deliver the right answers for those to the member.	As a member, I want to ask SEEA simple questions about my booked events, so that I don't have to search for the required information on my own in the app structure and different screens.	
1.4	Talk to SEEA	The built in assistant must be capable of displaying quick actions (open list of events related to user's input tag) in the suggestions bar related to the member's input.	As a member, I want to be able to get quick actions button to navigate me to the results related to the entered tags if any were available, so that I don't have to look for those in the Get Inspired page.	

1.5	Talk to SEEA	The built in assistant must be capable of receiving users input via text.	As a member, I want to enter my input to the assistant via text so that I don't have to speak out loud to it.	
1.6	Talk to SEEA	The built in assistant must be capable of receiving the users input via voice.	As a member, I want to give my input to the assistant via voice so that I don't have to write my interests (tags/topics) using the keyboard.	
1.7	Talk to SEEA	When the user enters the input via text, the built in assistant must give the answer via text as well.	As a member, I want to receive the output from the built in assistant via text when I enter my input via text, so that it adapts to my convenience.	
1.8	Talk to SEEA	When the user enters the input via voice, the built in assistant must give the answer back via voice.	As a member, I want to receive the output from the built in assistant via voice when I give the enter via voice, so that I don't have to read the written answer and listen to it instead.	
1.9	Talk to SEEA	When the user choses to enter the input via voice, the input must be displayed as text as well.	As a member, I want to read what did the built in assistant understood from my voice input, so that I can check if the built in assistant understood my input correctly.	
1.10	Talk to SEEA	When the built in assistant respond to the users input via voice, the output must be displayed as text as well.	As a member, I want to be able to read the built in assistant voice respond, so that I can rely on text in case I didn't understand the output via voice.	

2.1	Search for event	The system must be capable of retrieving a list of events corresponding to the users entered search criteria.	As a user, I want to be able to enter my search criteria, so that I receive a list containing events that matches this criteria exactly.	
2.2	Search for event	When no matches were found, a message must be displayed to inform the user that no matches were found.	As a user, I want to see a message indicating that no results were found in case no matched events to my criteria exist, so that I can go back and change my criteria.	
2.3	Search for event	When all fields in the search criteria are left empty, the system must display a list of all possible events.	As a user, I want to see the list of all available events in case no search criteria were entered, so that I explore all events that are available.	
2.4	Search for event	The system must check the date of event entered in the search criteria before proceeding to the results page to notify the user if the date does not match the expected format.	As a user, I want to be notified if my entered date for the event to search for is not accepted, so that I can change it to match the expected format.	
2.5	Search for event	If the save search switch is toggled on, the system must save the search criteria entered by the user.	As a user, I want to be able to save my entered search criteria, so that I can use it later in the future to search for events without refilling the search fields criteria.	
2.6	Search for event	If the save search switch is toggled on, at least one of the following must be entered: Search text, place, event type or/and event hosting organization.	-	

2.7	Search for event	The system must display the available options for the organization to the user to choose max. one of the listed organizations.	As a user, I want to see a list of the organizations available in the search criteria, so that I can choose one of them because I usually don't remember their names by heart.	
2.8	Search for event	The system must display the available options for the event type to the user to choose max. one of the listed event types.	As a user, I want to see a list of the event types available in the search criteria, so that I can choose one of them because I don't know what types are available.	
2.9	Search for event	The system must indicate where does the results list ends.	-	
2.10	Search for event	The system must show the most important information only on each event card in the results list.	As a user, I want to be able to read information about the event on the card of each event, so that I can decide which one to click on to see its full details.	
2.11	Search for event	The system must be capable of filtering the visible events list based on member's preferences entered in the filtering popup window.	As a member, I want to be able to filter the shown list of results even more, so that I can get results corresponding to my preferred saved or in-screen entered tags(topics).	
2.12	Search for event	The system must be capable of saving an event into the Saves Locker when the user clicks on the save event icon.	As a user, I want to be able to save events into my Saves Locker, so that I can find them much quicker in the future.	
2.13	Search for event	The system must be able to navigate the user to the default map app set on the smartphone when the user clicks on the map icon.	As a user, I want to be able to view the events location on my map app, so that I can check the distance to the event's location.	

2.14	Search for event	The system should be able to save events into the user's default calendar app under the correct date, time, location and event's name, when the save to calendar icon is clicked and the permissions are granted.	As a user, I want to be able to save events to my calendar automatically, so that I don't have to enter the events details and construct the event manually.	
2.15	Search for event	The system must show all event's details in the Event Details page.	As a user, I want to be able to see all event's details, so that I can decide whether to book or not.	
2.16	Search for event	The system must be able to book an event for the member, when the user clicks on book now button.	As a member, I want to be able to book events directly from the app, so that I totally rely on the app and avoid navigating to the Swiss Engineering platform.	
3.1	Get Inspired	The system must be capable of providing suggestions to events which correspond to the member's preferred topics(tags).	As a member, I want to be able to receive suggestions to events that matches my interest entered to SEEA, so that I can explore what events, which I didn't hear of may interest me to book.	
3.2	Get Inspired	The system must show why is an event being suggested.	As a member, I want to know based on which tag(topic) has been this event suggested to me, so that I can decide whether to keep receiving events based on this tag or not.	
3.3	Get Inspired	The system must offer the ability to remove tags directly form the Get Inspired page.	As a member, I want to be able to delete tags directly from the Get Inspired page, so that I don't receive events related to a tag(topic) anymore.	

3.4	Get Inspired	The system must offer the ability to view all event listed under a tag.	As a member, I want to be able to view all events listed under a tag, so that I can explore them all in one page.	
3.5	Get Inspired	The system must suggest to talk to the built in assistant if no suggested events were found in order to add more tags.	As a member, I want to be able to navigate to the built in assistant directly from the Get Inspired page if no events were displayed, so that I can add more tags(topics) to receive suggested events based on them.	
4.1	Profile	The system must show the member's name, email and address loaded from the member's account on the Swiss Engineering platform.	As a member, I want to able see my email, name and address loaded from the Swiss Engineering database, so that I can check if there is an error or update my data in case any has changed.	
4.2	Profile	The system must be able to display the member's photo in the profile page.	As a member, I want to be able to add my image to the profile, so that I can personalize my profile.	
4.3	Profile	The system must display a tab for the booked events in the profile page.	As a member, I want to be able to navigate to the booked events from the profile, so that I can check which events do I have next.	
4.4	Profile	The system must allow the member to choose whether to use the light or the dark theme across the whole app from the profile screen.	As a member, I want to be able to choose whether to use the dark or light theme, so that I can personalize my experience to make it match my own preferences.	

4.5	Profile	The system must show a list of all booked events when the member clicks on my reserved events tab in the profile.	As a member, I want to be able to see all my of booked events, so that I know what events are coming next.	
4.6	Profile	The system must show a message in the booked events page indicating that there are no booked events if no events been booked yet.	-	
4.7	Profile	The system must show a message indicating that the member has reached the end of the booked events list, if at least one event is booked and previewed.	-	
4.8	Profile	The system must be capable of showing the member the booked events page with the booked events (if any) even without an active internet connection.	As a member, I want to be able to see my booked events even offline, so that I can see my upcoming events.	
4.9	Profile	The system must group the booked events based on their dates in the booked events page.	As a member, I want to be able to preview my booked events grouped by their dates, so that I can find the information needed in a more convenient way.	
5.1	Saves Locker	The system must have a page containing both saved events and searches.	As a user, I want to be able to save events and searches, so that I can access them without having to enter the search criteria or search the events once again.	

5.2	Saves Locker	The system must separate both the saved events and searches from each other.	As a user, I want to be able to choose what to view on the screen whether saved searches or liked events, so that I can access the data needed faster.	
5.3	Saves Locker	The user must be capable of removing saved searches and/or liked events from the Saves Locker.	As a user, I want to be able to delete saves searches or events from my Saves Locker, so that it doesn't get crowder with events or searches that I don't wish to have anymore.	
5.4	Saves Locker	The user must be capable of viewing and deleting the Saves Locker elements offline.	As a user, I want to be able to see my data saved in the Saves Locker even offline, so that I can see and/or update them in all situations.	
5.6	Saves Locker	The system must show a message indicating that the Saves Locker for event or searches or both is empty in case it was.	-	
5.7	Saves Locker	The system must offer the user the ability to add items to the Saves Locker in case it was empty be offering a quick action button to save either new events or searches.	As a user, I want to be able to add elements in fewer clicks to my saves locker in case it was empty, so that I save the time navigating to the wanted screen using the normal hierarchy and steps.	
6.1	Personal community	When a member books a new event, this event should be suggested in the Get Inspired page to all other members who the member attended events with before.	As a member, I want to see suggested events, based on other members latest booked events, who attended the same events with me before, so that I can explore new events that might interest me.	

6.2	Personal Community	The system must indicate in the Get Inspired page that an event is being suggested because people with same interests booked it.	As a member, I want to be able to see if events have been suggested because other members, who attended previously an event with me, have booked it, so that I can identify that it is not related to my own entered tags and interests.	

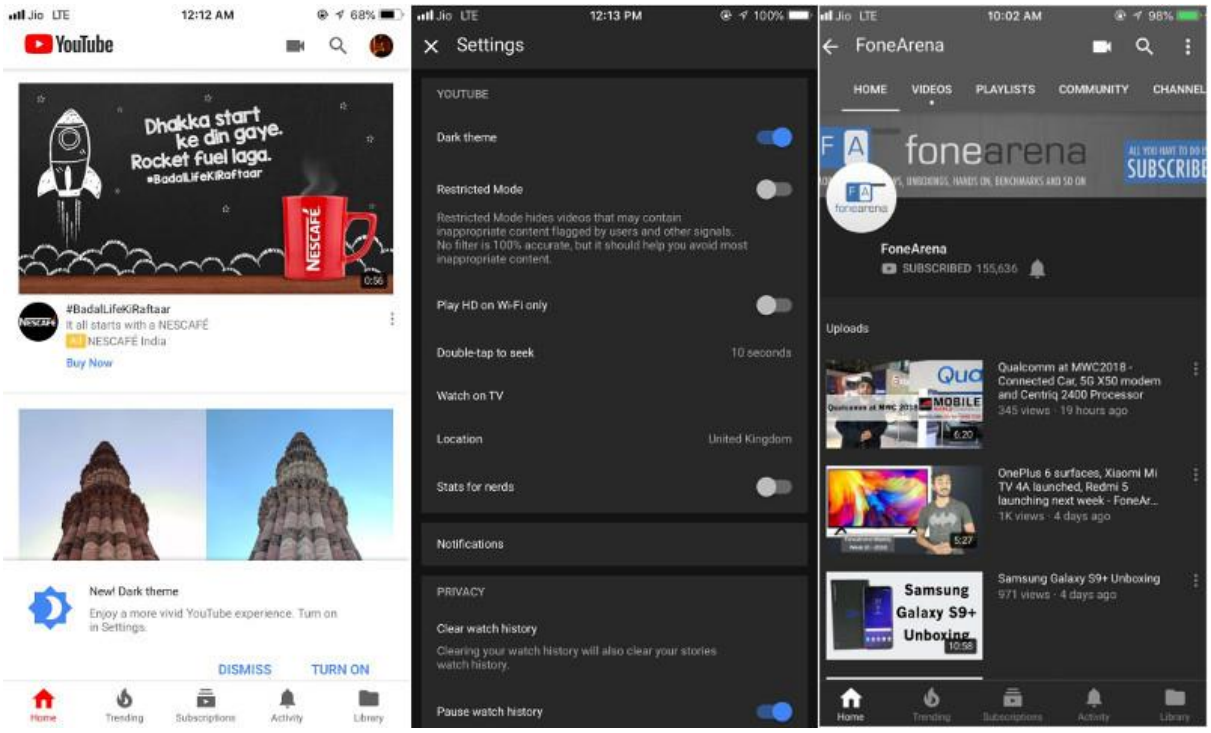
9.8 Design Story

This sub-chapter discusses the design story, the design inspiration and how did the design developed during the project lifecycle.

Design Inspiration

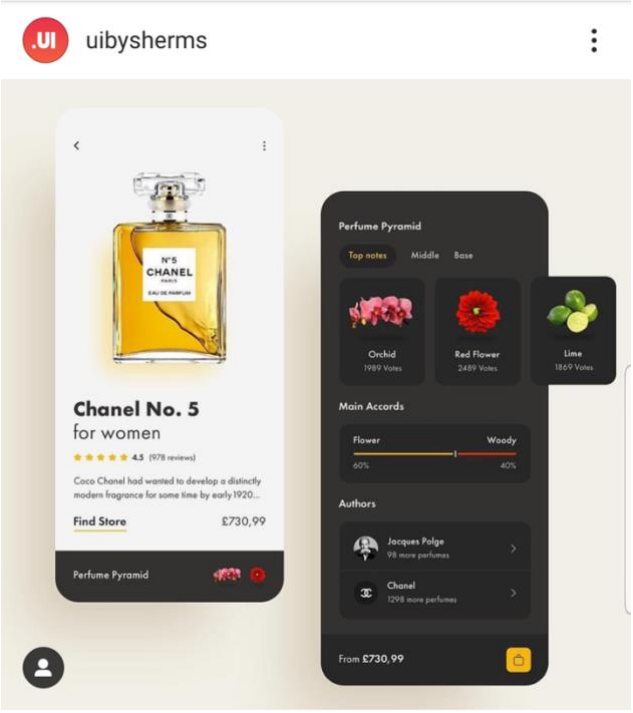
As we do live in Switzerland, the swiss popular apps that are heavily used by the swiss were our starting point. We had a look at the SBB app and Ricardo as we believe those two are the most heavily used in Switzerland and represent more or less the swiss culture when it comes to design apps.

But we didn't stop there only as we have also included Youtube, Facebook and also the Android Pie and iOS design concepts in our design plan.



Appendix Figure 10 Youtube App

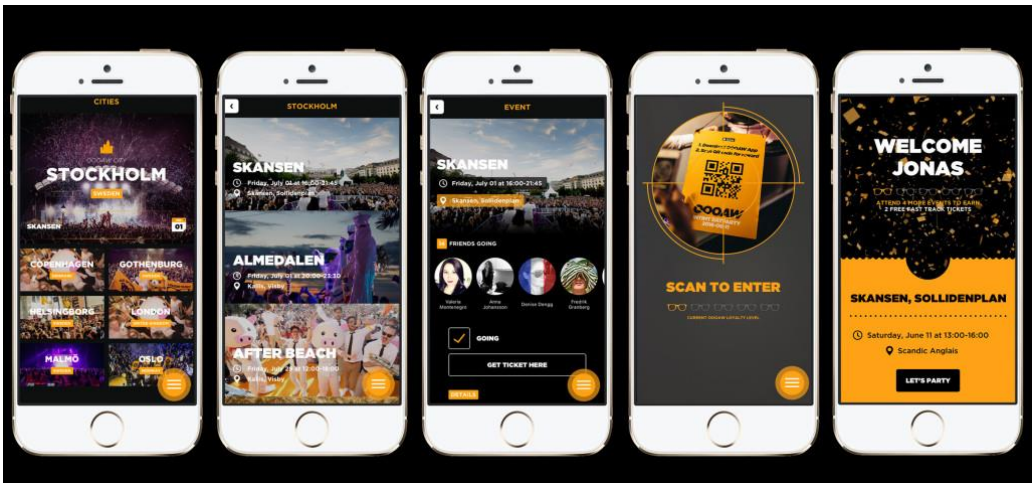
The underdog English app designer Alex Pesenka had made his way also into our list his design of the Perfum Pyramid app that as we thnk embraced more or less the design language that is applied on a globally known app and not locally.



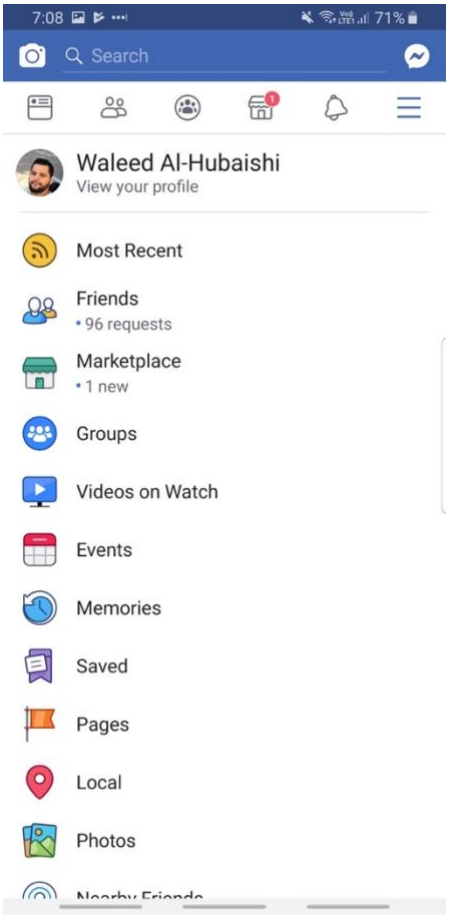
Appendix Figure 11 Alex Pesenka Perfume Pyramid design

Appendix Figure 12 VFW Event app, source: www.vfw.org/media-and-events/latest-releases/archives/2019/2/vfw-launches-mobile-app-ahead-of-2019-legislative-conference

Appendix Figure 13 Facebook event app, Source: the verge, https://www.theverge.com/2016/10/7/13192918/facebook-events-app-ios-android

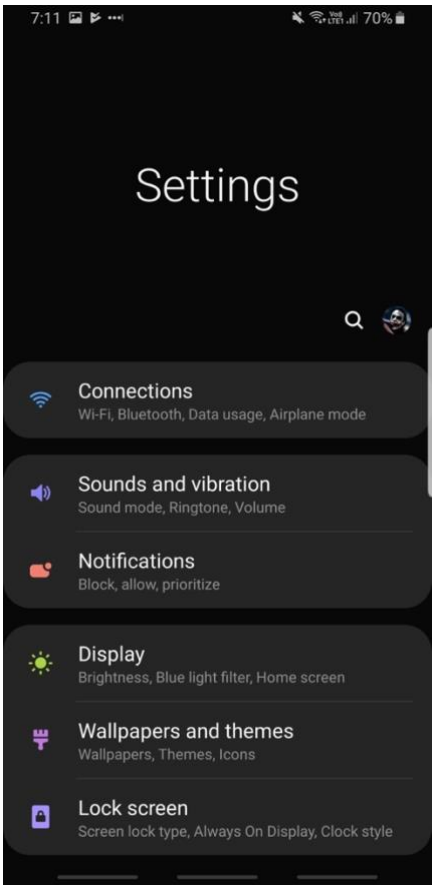


Appendix Figure 16 oooaw events app. Source: www.oooaw.com/mobileapp/



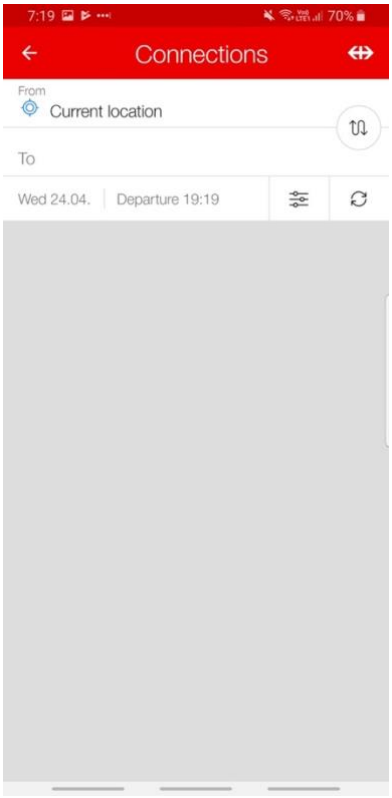
Appendix Figure 14 Facebook settings page

Appendix Figure 15 Facebbok news feed

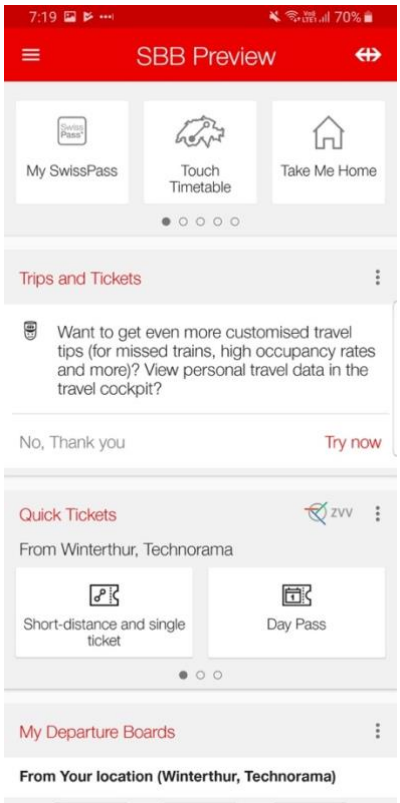


Appendix Figure 18 Samsung ONE UI settings page

Appendix Figure 17 iOS settings page

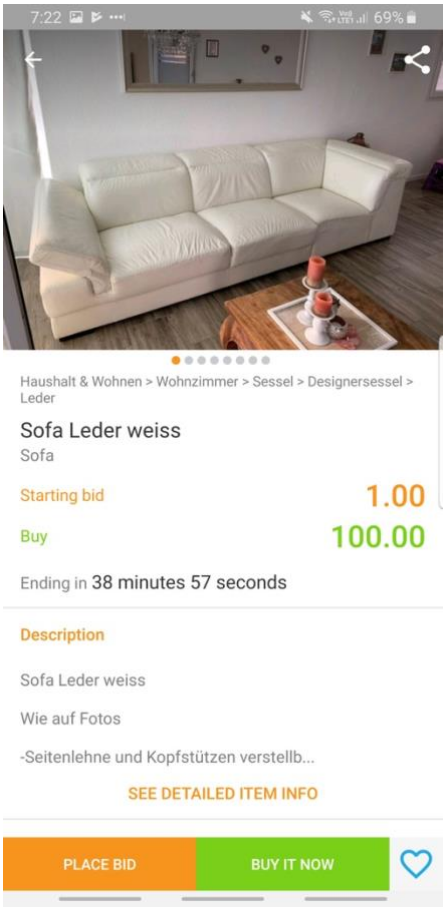


Appendix Figure 19 SBB search screen



Appendix Figure 20 SBB App home screen

After exploring all of those well-known apps, and given that they are well known then their design concept and language are more or less agreed upon locally and before all globally, we have decided to follow their foot prints and design our app in a way that embraces this design concept, internally in the screens of the features as well as the home screen.



Appendix Figure 21 Ricardo details screen

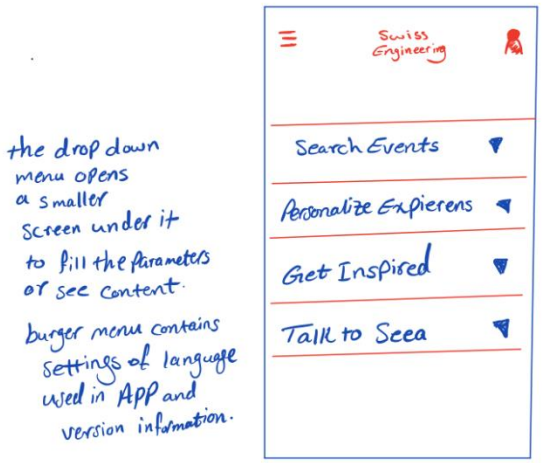
Information architecture

The app end-user according to our defined persona agreed upon is engineers, mid-thirties that come in general from different fields, specialties and have different technological affinities, starting from low. That was the main reason why we have decided to make the app features all listed in the home screen so that the user knows at the first glance what can be done using this app, and also why we insisted on making all relevant information regarding each feature represented in a way that is the most clear to the user and in a way that the user would expect.

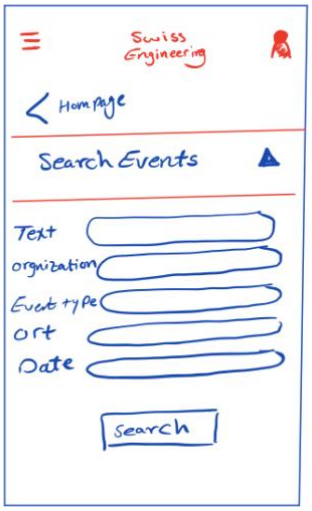
The way is it now implemented is more or less the same way that our inspirational apps presented their information to their users, as well as the web platform of Swiss Engineering, yet in an easier way to read and understand.

Lo-Fi Prototype

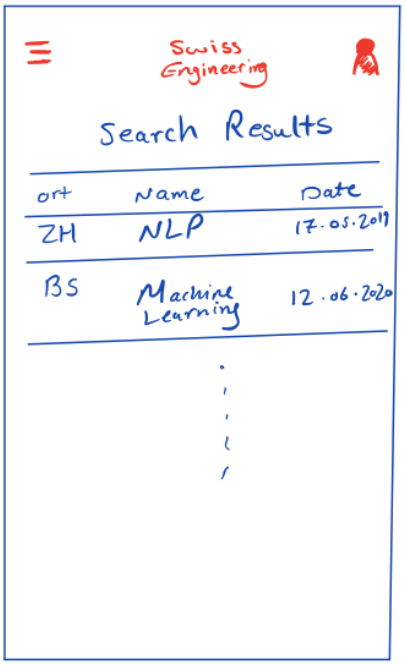
The team started asking a sample of the end user of what we would they expect to see in a home screen in an app of an event company, and we have also searched that online to know what do the apps offer in their home screens if the intent of the app is to reserve events. The result of searching and asking were translated into our first Sketch prototype.



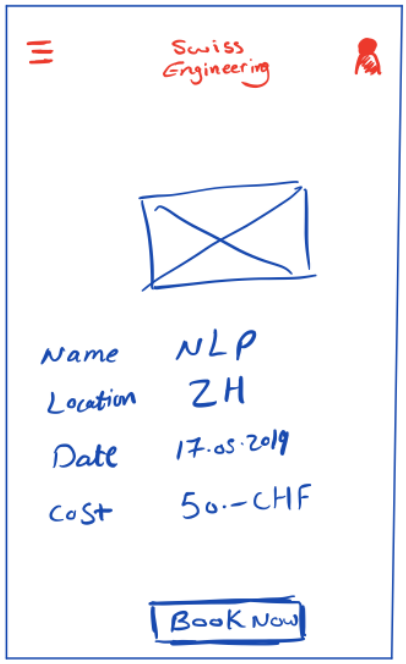
Appendix Figure 23 Home screen initial concept



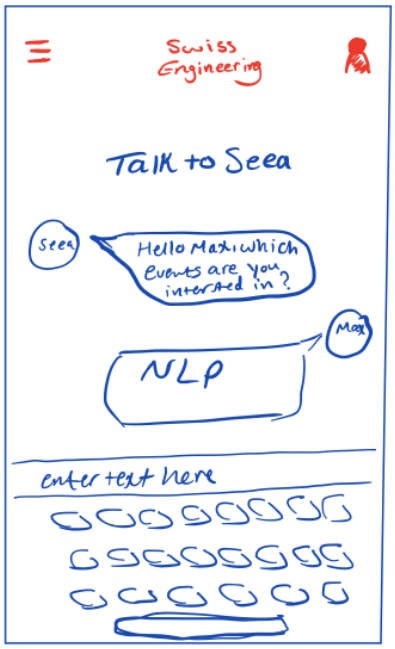
Appendix Figure 22 Search screen concept



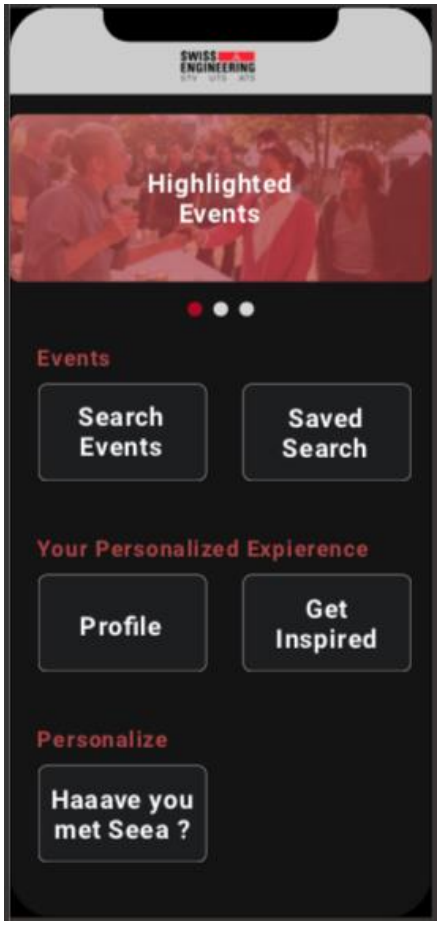
Appendix Figure 25 Search critieria entry input



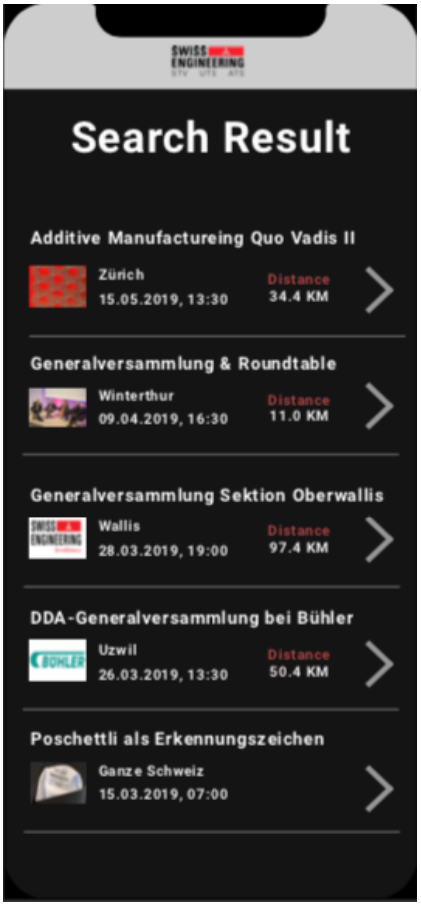
Appendix Figure 26 Event's detailes



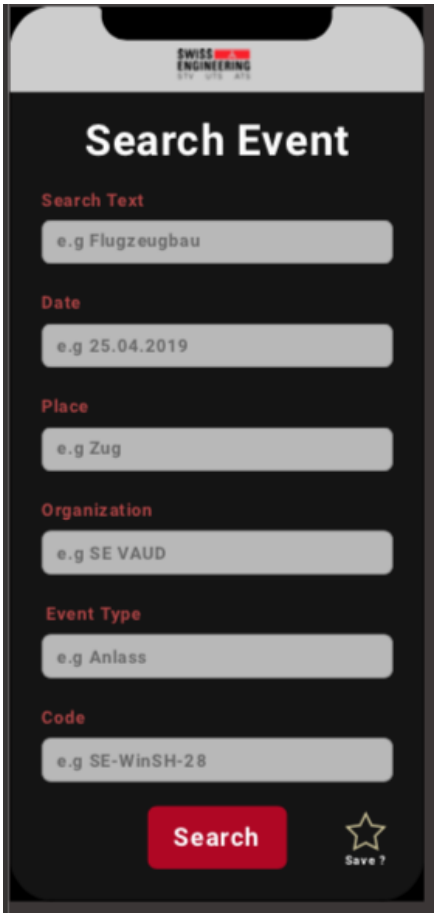
Appendix Figure 24 Interact with SEEA



Appendix Figure 28 Home screen 2nd concept



Appendix Figure 30 Search results 2nd concept

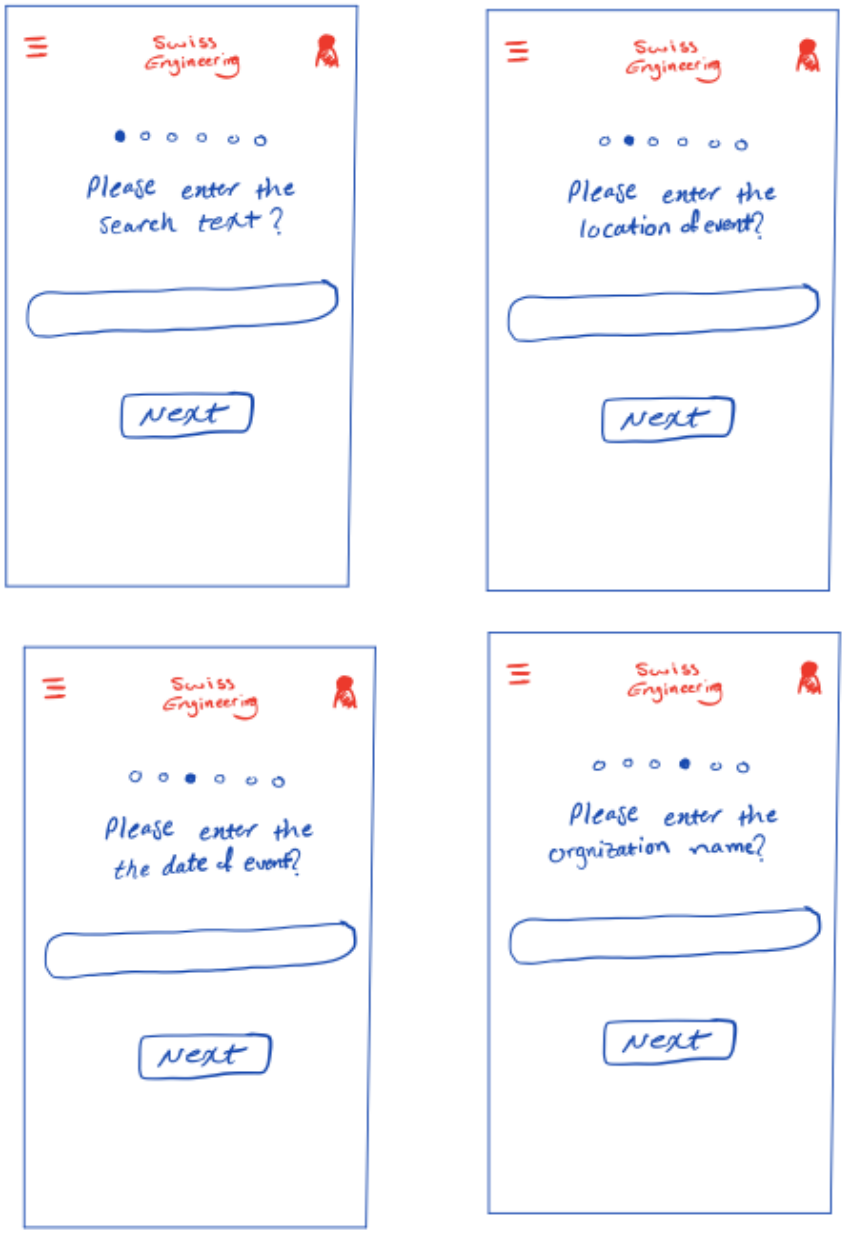


Appendix Figure 29 Search criteria entry 2nd concept

Appendix Figure 27 Online concept of a home screen of an event app

The user tesing were then done using the sketch prototype, and we have collected a lot of feedback, which address some issues regarding the color of choice and also the old school text fields meant to gather the input.

We have then tested another way to collect the user input to invoke a search query, with that the user should enter in each screen an input for the search query, but the user sample didn't like it much because they have said that it requires much time than a usual search, and we think what they have meant by much time is too many clicks to see the results.

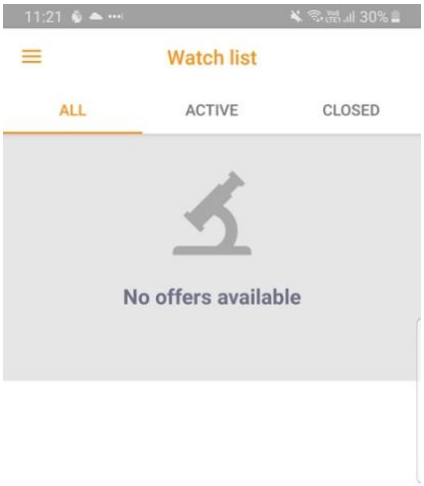


Appendix Figure 31 Gathering user input for a search query

Yet we didn't get a single feedback regarding the way the information are displayed in. but we got a designer feedback regarding the way the photo is displayed, he suggested that maybe masking the photo and making it full width would make it more consistent with the other elements that are shown in a table form. We have then used the same idea to display the highlighted events in the home page.

Edge cases

The table would be empty if the user doesn't have saved events or tags, and full if the user has many. A label can be displayed in the middle of the screen if the table would be empty indicating that there are no tags or events to display. That is absolutely fine as this is the way it works in the apps that we have looked at, and this is a fine compromise when such a feature needs to be included.



Appendix Figure 32 Ricardo app showing no results for a search qeuery with no matches

The design doesn't change also over time. Gamification is not a feature that we are willing to implement and keeping the UI the way it is from the first day and further is a key element to keep the end user familiar with the app. The most important thing to us is to not hide features deep inside so that user can't find them, and also to give a nice familiar UX UI to our end user so that the user keeps coming back without any fear that this a complicated app what so ever "keep it simple and easy".

Regarding the different type of users, we remember that you have told us to focus on the registered users only due to the scope as well as the rare chance that a normal user would this app on the store anyway "if he doesn't know Swiss Engineering the user would search for it on the app stores", but this app is only for registered users only and no registered user can access any of the features. In the future it might be possible to let a non-registered user use the app without offering the ability to book events or even save anything as there is no profile corresponding to the user.

Social Component

The social component is only a back-end feature, that will be built in and the user has no way to interact with it. We will ask the client if he want to give the user the possibility to turn it on or off, if yes, then this will be handled with a switch in the profile page.

Voice Assistant

The assistant role in our app is to get what the user is interested in and also answer a simple question like “when is my next event”.

It will not perform a search function cause 100% of the end user sample said that they will never use it this way as entering the text is easier and more convenient.

We think that placing the voice assistant in the home page was a key to make it visible to the end-user, and during our test with the end user they were all curious about what that button do, so they have clicked on it and got what it do at first glance, so we do believe that usability wise is ok considering what it meant to do.

Consistency

The dotted line is representing the number of the highlighted events, the user can swipe right to see the next highlighted event and by clicking on it a details event page would open.

The user can always go back by clicking on the arrow that will be placed on the top right of every screen, except home screen of course, it was our mistake not placing those in the designs.

Unfortunately we didn’t get what you have meant by the other four points, so we think an appointment would be perfect to clear all of those issues.

Usability

The fields can be all field or all empty, or partially. If all filled or partially then the most specific results will be displayed, if all empty then all events will be displayed. That is how it works on the platform and also in most of the apps with search function.

The live search update wouldn’t be needed in here because the search input is in a different screen and not in the same screen as the search results.

The search is not intelligent, it will search exactly what the user wants. The suggestion should be displayed in the Get Inspired page – which we didn’t include by mistake – which looks basically like the search result page. Those events are selected by the tags that the user

entered in the conversation with Seea or based on the events that people who the end user attended events with before (The Social Factor).

9.9 Project Plan

Both Versions of the project plan can be found in the Administration folder inside the Git Repository.

The Folder Project Plan inside the Administration folder contains the plan for the first part of the project between February and June, and the second part between July and August.

9.10 Online Web Summary

To browse the online summary of this project please click here. Alternatively, the following link can be pasted in the browser's URL bar to view the page.

Link: https://web.fhnw.ch/technik/projekte/i/ip519/IIT16/index.html

9.11 Gitlab Repository

The following link can be used to clone the Git-Repository.

Link: https://gitlab.fhnw.ch/waleed.alhubaishi/ip5-touch_und_sprachgesteuerte_event_app.git

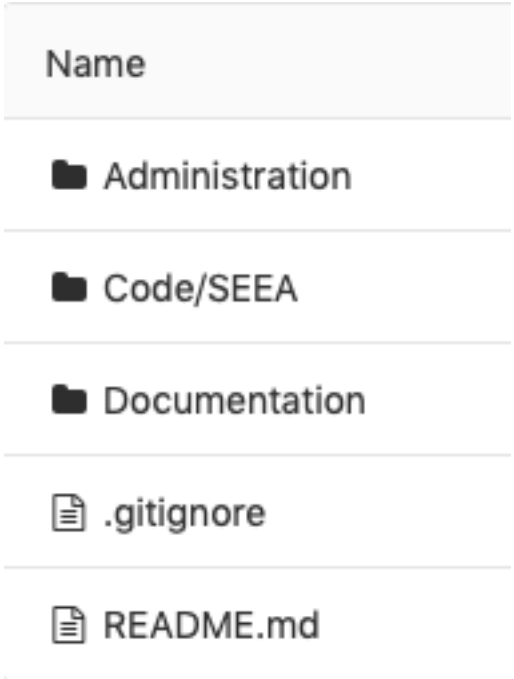
This Git Repository contain two main branches. The first branch is called Master which contains the final version of the application in addition to all the documents.

The Administration folder contains the Meeting Protocols and previously mentioned Project Plan.

The Code Folder contains the final version of the Application.

Documentation Folder Contains the Design Folder with all Mockups and brainstormed ideas, as well as the Sketch project that contains all the mockups and designs, and the Design Story in text.

The second branch is called Develop, and it was used to develop the application before merging its final version with the Master branch.



Appendix Figure 33 Git-Repository Structure

The README.md file for the application can be found inside the Code/SEEA folder.

10 Honesty Policy

I declare herewith, that the attached Project documentation was written only by me independently, without the help of third parties, and only using the specified attached sources.

Name: Waleed Al-Hubaishi

Place, date: Brugg, 16.08.2019