Welcome to this Tribefire Tech-Demo, which will show Tribefire's capabilities along Braintribe's Playbook by exploring the Proof-Stage where we build a POV - a Proof-of-Value - by setting up a first platform use case with simulated data and services.

The fictional company we deal with already uses Tribefire as a strategic platform and already built a few cases.

Now they want to create an internal case around their sales process with a pro-active overview for their C-level suite and a mobile solution including a gamification feature for their sales staff.

With Tribefire we follow a top-down approach, so there is always idea-first and business-first. Therefore, in the beginning the involved roles are:
The head of digital
Business leaders
Business analysts

We start with exploring the Asset Library to see what is already available and getting inspired how the business idea can be achieved.

For example, we can inspect already built applications, like this one, which provides a comprehensive overview for call center agents…

…or a network-based view onto doctors and patients showing the of data from different angles to get additional insights in terms of relations and context…

…or an app which allows to browse through a company's customer landscape in an explorative to find and inspect respective opportunities and other related data and documents. Those are connected to the specific cases and can be navigated in a logical way but are fetched from disparate sources of data and services. This seems to be a good basis to start with!

So, we explore what assets were used for this case and find that we don't have to start from scratch but can reuse available building blocks, which we call *assets*.

One asset that looks like a candidate for being reused is the Sales Model.

We can browse through the semantic objects, see how they are related and inspect the details of Company, Customers, SalesOpportunities, Offerings and the involved Persons.

This model is obviously a good starting point. So, the business proceeds now by working together with the Data Engineer, who will import the model in order to extend and re-use it.

To do so, we have different options - in our demo case we use a so-called exchange package, which is uploaded and then imported.

Now we can work with the model and adapt it to our needs.

As required by our use case we can add our own elements with their specific properties. For the demo we create an element called DemoCompany with one property, the demoProperty.

This was actually pretty easy, and to make it really meaningful and also to assure the compatibility we can reuse the existing Company element. This can be achieved by inheritance.

Also, we can add additional models and reuse their capabilities, like for example to organize our company cars.

We just add the required model to the one we are working with, bring the needed element on the screen and connect it accordingly.

You can see, this *reusing* is important in tribefire, as it speeds up the process drastically.

After we're ready with the modeling part we want to bring our model alive. To avoid the technical integration efforts in this early stage of the project we run the model in simulation mode.

To do so, we use a so-called Access. Such an Access can on the one hand connect to a data source, which is in our case just *simulated*, but on the other hand it provides the model as its own API. This API provision usually means manual efforts. With TF this is generated automatically - no code is involved here.

One way of using the API is the TF Explorer, which is a powerful generic client that is part of the platform and can deal with data defined by any model.

Here we can create our data sets. In the Proof-stage we deal with simulated data, but working with data from connected systems in a later project stage will feel exactly the same.

To speed up the process we can also import data from Excel or CSV files. We upload the excel file with the data and trigger the import on our access and model. Afterwards we can further work with the imported data.

Such, we can build our use case right away.

By deploying the Access the Model-API was generated automatically.

The APIs are available via Java, JavaScript, Stream and REST, which can for example be accessed via Swagger UI, where we can also try it out directly, as TF created the essential Http-Methods for every model element.

We can perform the GET call for our Demo Element type and inspect the result as JSON object immediately in the UI.

This works of course in the other direction, too. Via the API we can easily change an existing value, which can be inspected afterwards in the Explorer.

Now it’s time to upload our created assets to the Asset Library so all involved parties can see the progress and review what we have achieved so far.

Here we can show the data in the explorer which is now customized for according to the use case. Also additional demo was imported, so the data model is more tangible now.

Also, just by configuration without a single line of code, the data engineer can set up a data Xperience with TF’s reusable UI components to provide the modeled data in a meaningful and useful way. Thus, we can browse the data via network view…, in lanes that provide groups of datasets… or along a timeline and other ways.

Now it’s time to involve the UI development team. In order to implement the required applications, they use the Tribefire REST API we showed before as well as the just demonstrated pre-built UI components.

The first app they implement is built for members of the C-level suite. Via alerts, based on sales-related events one gets informed pro-actively and can then drill down to get a comprehensive overview of the situation, which is orchestrated by data from CRM and ERP systems, news feeds and document management systems.

Via calendar invitations, email functionality, and other means they can react immediately.

The second app is built for the sales staff.
It provides a gamification feature which encourages salespeople to maintain their sales data, which can be achieved easily in a mobile and user-friendly manner.

We reached the end of the technical Tribefire demo. We showed how we use the platform to build a POV. The next step in our playbook is the MVP where we provide a first live-version of your platform.

Thanks for watching.