AppNavi

Product description

AppNavi advises its customers on process optimization, the digitization of these processes using tools or platforms developed in-house - and trains users to use digitized processes efficiently.

Date: 18.04.2021

AppNavi GmbH

Management:
Carsten Neumann

Atelierstr. 29 · 81671 Munich
Tel.: +49 89 217 505-0
Fax: +49 89 215 80 407
info@appnavi.eu

Commercial register:
Local court Munich HRB 216915
Tax-ID: 143/114/50124
VAT-ID: DE298859635

Bank details:
Stadtsparkasse München
IBAN: DE22 7015 0000 1003 6209 43
BIC: SSKMDEMMXXX

Content

1 About this product description

Thank you for your interest in AppNavi. With this product description we would like to give you an overview of the functionalities of AppNavi Since we are continuously developing Appnavi to meet the needs of our customers, there may be temporary differences between the description and the product - we reserve the right to make changes in the description as well as the product accordingly.

2 What is AppNavi?

AppNavi is the "easiest to use digital adoption platform". With AppNavi, any user can use any software correctly right away. Without training. Without queries.

Integrated within the respective target system, it guides users along a defined use case step by step through the respective live application. At each point of a route, tool tips provide assistance for the user to correctly execute the next step.

In addition to the routes available for this purpose, three other powerful functionalities are also available: Hotspots, Learning Collections & News. The latter allow, for example, to actively inform about process changes. In this way, new or changing processes can be learned and executed directly in the live system simultaneously with productive work.

2.1 Benefits through AppNavi

Especially in complex or rarely used applications, users often feel lost. As a result, a lot of time is wasted through unnecessary training, futile attempts or expensive inquiries to colleagues or the service desk.

This is where AppNavi helps by empowering, supporting and accelerating users. Employees are trained "on the job". This reduces training costs because training occurs at the time of the problem, rather than targeting fleeting stock knowledge. Necessary tool knowledge & process knowledge is made available "just in time" and ensures:

- Improved acceptance & higher user satisfaction
- Fewer user errors and higher user efficiency
- Higher ROI of the target application
- Reduction of maintenance efforts (self-adapting routes, with new releases of target application)

2.2 Supported applications

AppNavi can be integrated into any browser-based software. Depending on the integration strategy, both in-house developed applications and SaaS solutions come into question here. Here are some examples:

- **HR**: SAP SuccessFactors, Workday
- **Collaboration**: Teams, SharePoint, other Office 365 modules, Jira, Confluence
- **BPM**: Jira, Ultimus CPS, Service Now
- **CRM**: Salesforce, Aurea CRM
- **Dashboards:** MicroStrategy
- … and many more

2.3 Supported customer situations

AppNavi distinguishes three typical customer situations:

- Enterprise customers
- SaaS provider

- Training providers or consultancies

AppNavi supports users in web-based software. The more complex this is, the better the strengths of AppNavi can be played out. The classic use case for AppNavi is therefore a company that wants to support its internally or externally used application landscape. Employees, applicants, customers as well as interested parties of this customer company are guided through the systems with the help of AppNavi.

In addition to the companies themselves, providers of Saas solutions can also use AppNavi. Here, the support is created centrally by the provider and delivered to all of the provider's customers.

In addition to SaaS providers, training providers or consultancies can also create content for selected applications and play it out to their customers.

In the latter two customer situations, in addition to the centrally provided content, individual content can also be created and delivered for each customer.

2.4 Integration strategies

AppNavi is integrated into an existing browser-based software on the client side. This is done centrally by the customer's IT department. End users do not have to do anything - AppNavi becomes visible to them without any action on their part. With regard to the integration of AppNavi, two strategies can be distinguished:

- Direct integration
- Integration per Browser extension

Both strategies lead to a client-side integration of AppNavi into your software. With direct integration, a JavaScript reference is embedded in the target application. For self-written applications, this is usually the case. SaaS solutions also offer such an integration option in a large number of cases.

With direct integration, the integration is done application by application. If you want to use AppNavi within an organization in different applications at the same time or if you do not have access to the target application, integration via browser extension is available. This enables the browser to use AppNavi within an application.

2.5 Compatibility

AppNavi works as a navigation system for all browser-based applications. It is completely irrelevant on which device type (laptop, tablet or cell phone) or with which operating system (Windows, Mac OS, Linux, etc.) your users want to use AppNavi. For mobile devices, adjustments usually have to be made, which are taken into account in the initial integration project.

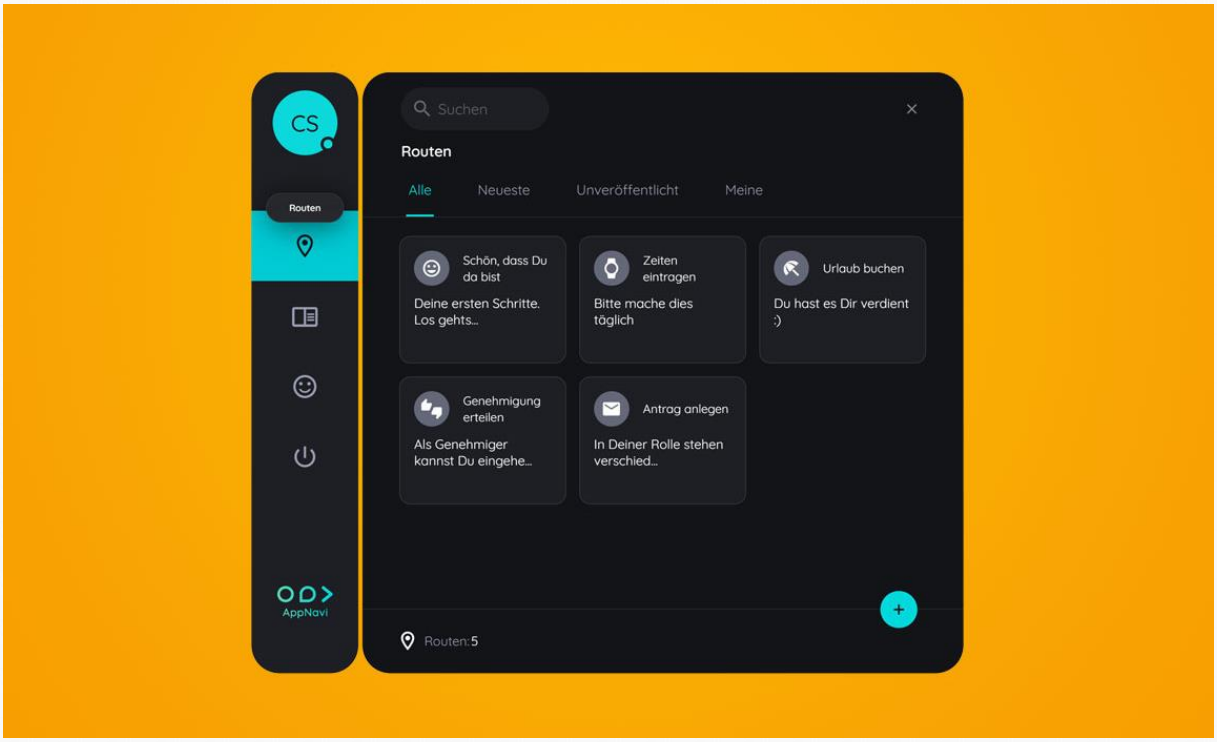
AppNavi's object detection works on HTML and SVG elements as well as in all underlying frameworks (Vanilla, React, Vue, Angular, etc.).

2.6 UI components

AppNavi customers are mapped in a so-called tenant. All your apps, users, content, etc. can be managed in this tenant. The tenant can be accessed via the **customer portal**. This is a website that can be accessed via the browser.

With the help of a created app, the **AppNavi widget** becomes visible within the target system. Our widget is the visual presentation of AppNavi for your users within your systems.

Depending on the role, a user can open AppNavi's **planner & recorder** within the widget. This is available for editing and creating routes. With the help of a graphical user interface, authors can create routes **very easily and without IT know-how**.



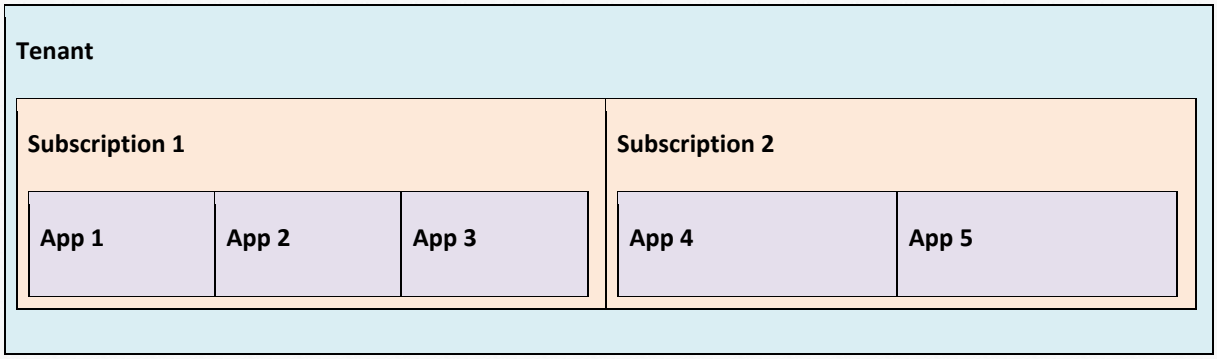
2.7 User types

AppNavi distinguishes between end users and authors. End users are supported by AppNavi and can use the content provided by authors. Thus, when using AppNavi, there is a large group of end users and a small number of authors.

Authors have a login to AppNavi. This can be used both in the AppNavi widget (within your target system) and in the AppNavi customer portal. End users are completely anonymous and do not need an AppNavi login - for them AppNavi appears when they move around the target system.

3 Architecture

AppNavi has a multi-tenant architecture. The topmost unit is the so-called tenant. Subscriptions are available within a tenant, which in turn contain apps.



3.1 Tenant

Each customer is represented by a tenant. Tenants consist of subscriptions, apps and contents.

Subscriptions represent organizational structures within the customer. Apps are the applications supported by AppNavi and Contents are the content provided there, such as routes or news.

3.1.1 Settings

Each tenant has different settings. In addition to the option to name the customer and the tenant, the following options are available for our end users:

Enable User Analytics

AppNavi has a strong analytics component. This enables data on the usage of AppNavi itself, the AppNavi content as well as the target application to be determined and transferred anonymously to the admins on the customer side.

The analytics functionality can be activated application by application - provided that this option has been activated for the tenant.

Enable Custom Scripts

Within individual applications and individual steps of a route, customizations can be stored in the form of JavaScript and CSS. Against the background of internal IT security, this option can be deactivated in the tenant if this is not desired by the customer ISOs.

3.1.2 Media library

Each tenant has a media library. This can hold images in PNG and JPEG format. Images uploaded in the route planner are stored in this library and receive a publicly accessible link. The media library fulfills two objectives:

First, it allows images used in routes to be visible to any user. For example, if you want to use an image from an internal SharePoint in your routes, this would not be visible to other users - by uploading to the media library, you ensure continuous availability.

Second, images go through a virus scan when uploaded to the library. This ensures a trustworthy use of the images.

3.1.3 User management

All authors within a tenant are created as users. Each user can then be assigned subscriptions to which they have permission.

Subscription owners and contributors only see colleagues of their own subscription(s). This enables the use of AppNavi within different organizational sub-structures. Tenant Owners can see and edit all users of the tenant.

3.2 Subscriptions

Subscriptions are used to build logical structures within the tenant. Depending on the client situation (see chapter “Supported customer situations”), there are different examples of possible structures:

- Entities or Business Units
- Locations
- Individual projects
- End customers of the AppNavi ISV
- Etc.

Subscriptions have a name and can be assigned users. For each subscription there is a group of owners and a group of contributors. A single user can therefore have different permissions to different subscriptions.

3.3 Apps

To display AppNavi in a new system, the URL of the system is first added to the tenant as an app. Both absolute and relative patterns can be specified here:

- https://your-URL.com/ (AppNavi is only displayed on the home page)
- https://your-URL.com/page1/ (AppNavi ist only displayed on page 1)
- https://your-URL.com/* (AppNavi is displayed on all pages of your system)

3.3.1 Settings

Each app has a number of settings. First of all, the widget features can be enabled or disabled. By default, routes and news are enabled as features within the AppNavi widget. Optionally, the collections can be switched to it or the stanadfeatrues can be disabled. Authors can thus configure the widget according to the requirements in their system.

In addition to these settings, our CSP Fix feature is available. This comes into play if the target application prevents the integration of external applications via content security policy. By activating this feature, AppNavi can also be integrated under the CSP of the system.

3.3.2 Customizing

Apps können hinsichtlich ihrer Funktionalität und ihres Aussehens customized werden. AppNavi bietet hierfür ein Custom Code Modul an. Sowohl das vom Autor zur verfügung gestellte CSS, wie auch das JavaScript werden zur Laufzeit in den Scope der Seite integriert und ausgeführt. Dies geschiet vor dem Start von AppNavi, sodass Customizings bereits greifen bevor das Widget sichtbar wird.

3.4 Contents

AppNavi provides you with various contents to support the use of your systems. Contents are initially assigned to a subscription within the tenant. The assignment of a content to an app defines the editability of the content by the users of the tenant. Only the users assigned to a subscription can edit the contents. However, contents are visible across all tenants.

3.4.1 Versioning

Saving a content creates a new version of the content. For each version, the date of the change and the name of the changing author are saved.

3.4.2 Assignment to an App

Contents can be assigned to one or more apps. By assigning them, they become visible to all authors within the AppNavi widget. For end users, the content is not yet visible at this point, so authors can work on the content in the background.

3.4.3 Publishing within an App

To make content visible to end users, the current version of the content can be published.

Subsequent versions of the content become visible to end users by republishing. Unpublished intermediate versions remain invisible to end users.

3.5 Roles & permissions

To enable appropriate content management, the following roles are available to authors:

- **Tenant Owner**: Can manage subscriptions and content of all subscriptions
- **Subscription Owner**: Can manage content of one or more subscriptions
- **Contributor:** Can create and edit content

4 Routes

The central content type is routes. Routes guide your users along a defined use case step by step through the respective live application. At each point of a route, tooltips provide help for the user to correctly execute the next step in each case. In on-boarding scenarios, routes can also be assembled chapter by chapter and made available to users. So-called route collections can be used for this purpose.

Your end users can find routes in the AppNavi widget under the heading of the same name.

4.1 Representation

Within the AppNavi widget, users are shown all routes relevant to them in a menu. Within this menu, routes can be selected and started.

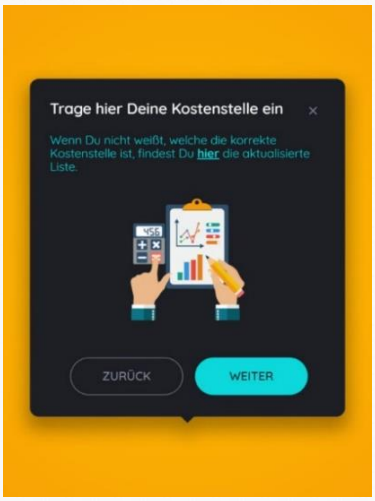
Routes are displayed sorted by the last modification date or based on an order defined by the authors. In addition, routes can be searched and found using a full text search.

4.2 Structure & components

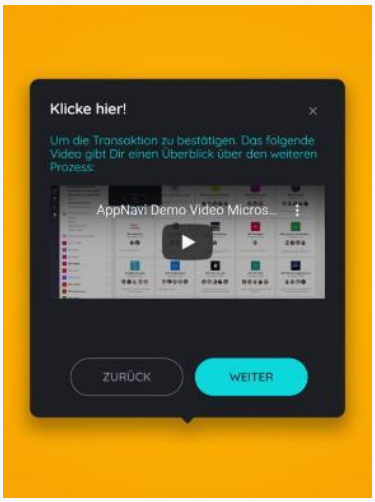
A route is identified by a title, a description and an icon. The content of a route is defined by a string of individual steps.

Two types of steps are available within AppNavi. One is intermediate information, the other is information linked to elements. Both step types are represented by tooltips. These consist of a heading and a content area.

Tooltip with text



…with an image



…with a video

Tooltips can be customized in size and shape as well as display different content:

- Text (WYSIWYG editor)
- Images
- Videos
- Iframes
- Links

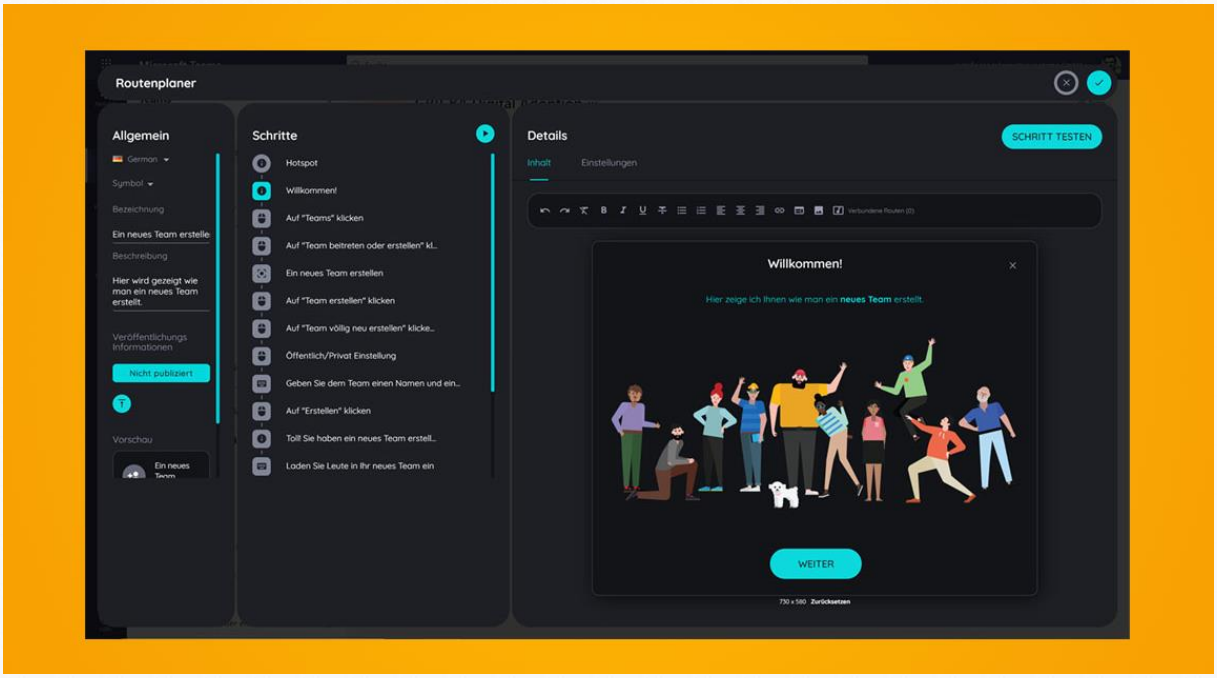
With the help of these options, tooltips can be designed very individually - in addition, existing content can also be used within routes, e.g. with iframes etc. To get from one step to the next within a route, six options are available in total:

- **Enter a value** within the application
- **Left click** within the application
- **Right click** within the application
- **Click Enter** within the application
- **Hover** within the application
- **Highlight** within the application - click on „**Next**“ in the tooltip

The last option lets you choose whether the highlighted area is clickable by the user or not.

4.3 Creating routes

The route planner is available for creating routes. This makes it possible to create routes very easily with the help of a graphical user interface. Technical understanding of the underlying application is not required.



Routenplaner

4.4 Element detection

The elements for a step are recognized with the help of a fuzzy score algorithm. This reliably identifies elements within a page based on a large amount of information describing that element.

The special feature is a high reliability. In contrast to common approaches, AppNavi does not store CSS selectors, but contextual information about an element. This information is automatically recorded and can also be adjusted by authors if necessary. Thus, elements are found, for example, even with dynamic IDs, etc..

4.5 Triggers & modes

Routes can be started in a total of four different ways:

- Click on the **tile** in the route menu
- Via **URL parameters**
- Click on a **hotspot** (small "sticker" on an element)
- **Program controlled**

With regard to the modes, two variants can be distinguished. On the one hand, a route can be started in learning mode. Here, all tooltips are run through step by step. The user clicks and taps in the application and thus progresses.

In addition, AppNavi has a speed mode. This allows previously automated steps to be clicked or text typed automatically through AppNavi.

4.6 Features

A number of features are available for routes within AppNavi. Here is a selection:

Automation (RPA): All (or individual) steps within a route can be automated. In this case, clicks or keystrokes are automatically executed by AppNavi.

Segmentation: To ensure that each user only sees the content that is relevant to them, the visibility of our content can be controlled using segmentation. The logic for this can be freely defined - common examples are a user's role or location affiliation.

Multilingualism: routes can be translated into different languages. This is possible both in the route planner and in the customer portal.

Automatic translation: A total of 25 languages can be translated automatically (at no additional cost) using an DeepL integration.

Linking routes: Routes can be linked automatically (based on rules) or by user feedback. Thus, even very complex non-linear processes can be easily mapped.

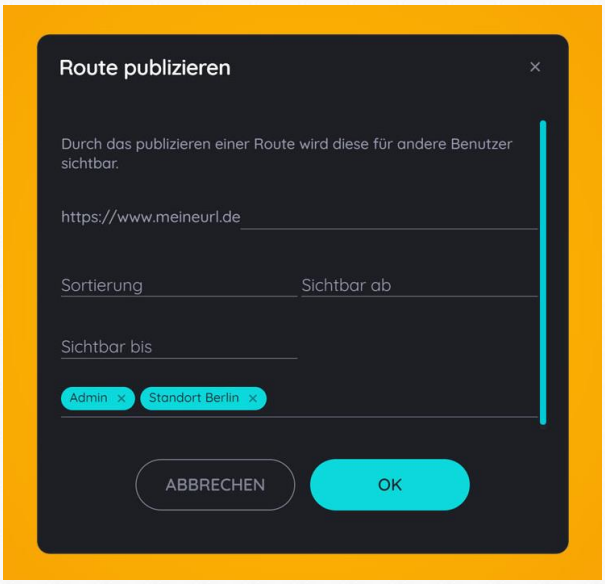
Responsiveness: AppNavi detects the resolution of the application and adjusts the appearance accordingly - even when window sizes change.

Auto Scrolling & Tracking: If the next element is not in the visible area, our algorithm scrolls to the right place: Horizontally and vertically, in the page as well as in iframes.

4.7 Versioning & publication

AppNavi has a two-step publication mechanism. A route created by an author can first be saved and thus made available to the other authors. Each saving creates a new version of the route.

As soon as an edited version has reached a certain maturity, the route can be published. This makes it visible to the users in the current version. In the background, the authors can continue editing and create new working versions, which can be made available to the users by publishing them again.



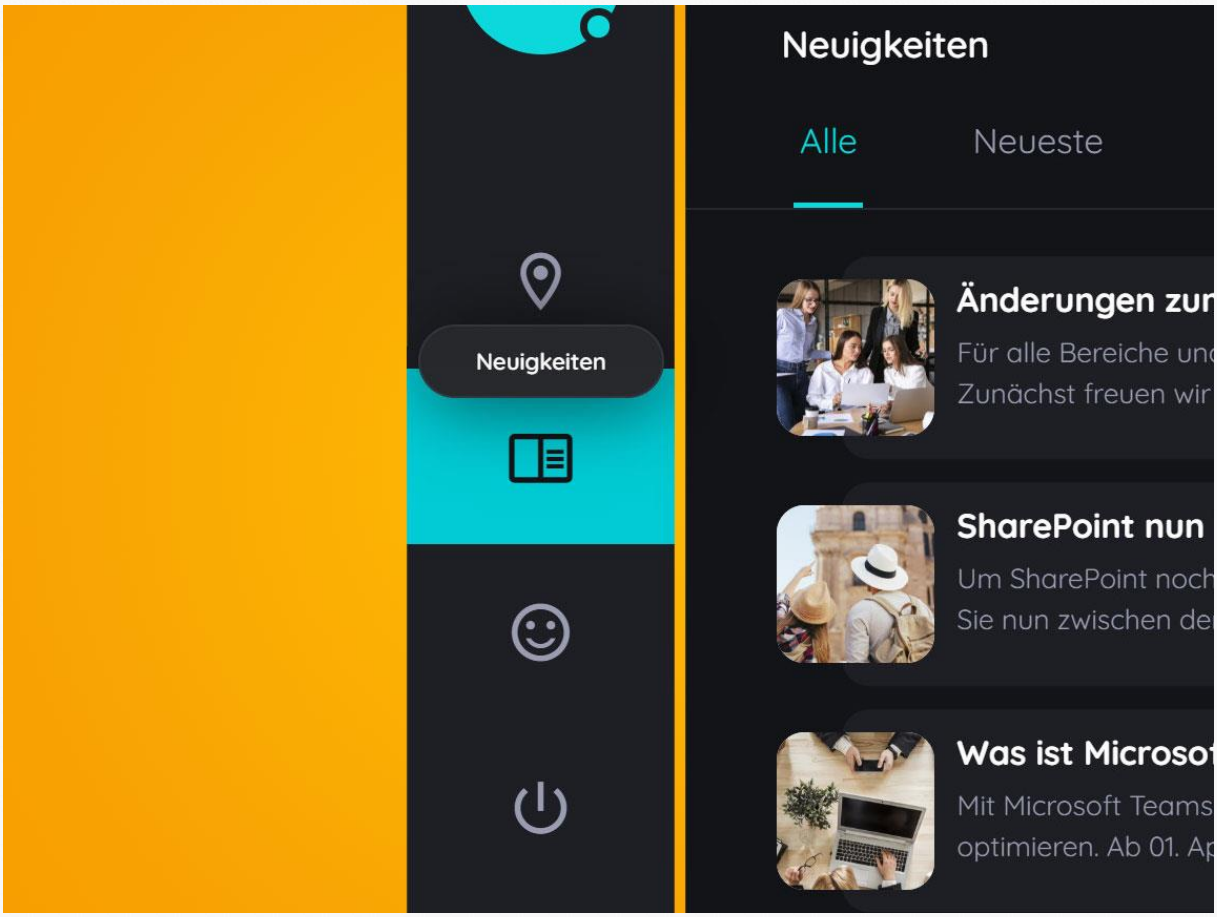
5 Other content types

AppNavi provides other content types besides routes. Depending on the customer situation, the focus in using AppNavi is to support, inform, accelerate, etc. News and collections therefore complement the feature set accordingly.

5.1 News

News, for example, makes it possible to display small articles on interesting content in a targeted manner for users. Here, too, segmentation takes effect so that the articles are only displayed to the relevant users and only at the relevant place in the system.

News can be initiated both from the AppNavi widget and as an announcement when the page is visited for the first time.



5.2 Collections

A feature from the OnBoarding area are collections. Collections allow routes to be combined into chapters. A collection can in turn be assigned to an app and then published. Users can run through routes within the collections and see the progress. Once all routes in a collection have been traversed, the collection is marked as complete.

When a collection is assigned to an app, all routes within it are also assigned to that app (if they are not already). Conversely, when a collection is removed from an app, the routes remain assigned.

When a collection is deleted, the user can specify whether (all) routes in the collection should also be deleted. If they are deleted, they will be deleted completely - with all their further assignments and publications in any other apps. If the user denies this, the routes remain available in the respective apps with unchanged assignments and publishing. The collection itself is subsequently deleted in both cases (including assignments and publications).

When a collection is published, all routes in the collection are also published. The following cases can be distinguished here:

1. one route of the collection is already published
2. one route of the collection is already published but currently invisible
3. a route is not yet published
4. a route is depublished (while it is published in a collection) in the app
5. the collection is depublished

Ad 1: The route appears in the route menu and in the collection.

Ad 2, 3, 4: The route remains invisible in the route menu, but appears in the collection.

Ad 5: The collection becomes invisible. Routes that were previously visible in the menu remain visible - all invisible routes remain invisible.

6 Analytics

AppNavi provides end users with various types of help when using new or existing systems. In order to provide the authors of these helps with better insight into how their content is being used, and thus improve the quality of the help, AppNavi provides an optional Analytics module.

First, analytics can be enabled on a targeted basis at the tenant level as well as at the level of an individual application. This opens the communication channel between the client and our Analytics module. In order to conserve the network connection, the Analytics events are collected in so-called event bulks. A bulk is simply a collection of events. These are then sent anonymously.

Here is a selection of the available events:

AppNavi widget

- How often is the avatar displayed?
- How often is it clicked?
- … and many more

Contents

- How often was which content (e.g. a route) started?
- How often was which step within a route run through and how long did this take?
- How often was a route interrupted at a step?
- How long is a news item read on average?
- … and many more

This information on the events subsequently provides a very detailed insight into frequencies, average durations, etc.

7 Data privacy

Basically, the functionality of AppNavi is based on a "data-blind" concept. I.e. AppNavi routes are started in the target system and know there the context of the elements to be found (element type, attributes, classes, approximate position, etc.). However, the content of the elements is completely unknown. Thus no input values, texts, or the like are stored.

At runtime, the fuzzy score algorithm determines the element with the highest match between recorded route definition and current screen situation.

The functionality of AppNavi (including the segmentation logic) is always executed on the client side - a transfer of information to the backend of AppNavi does not take place at any time.

8 IT security

As a SaaS solution, AppNavi is regularly tested by external pentests. These are based on common standards such as OWASP and check criteria such as session management, file upload and download, cross-site scripting, SQL injection, path traversal, parameter manipulation and other OWASP top 10 vulnerabilities.

In addition to the audits initiated by AppNavi itself, AppNavi was also tested in numerous other tests during customer implementations.

Authorization concept

- Access level according to the **need-to-know principle**
- User authorizations are defined and assigned by the respective departments
- Dedicated user roles are available

Data privacy

- Data is always encrypted - during storage ("at rest") as well as during transmission ("at transit")
- The integrity of the information is ensured by logging and version control of all user changes
- The data is stored in an AWS data center (Frankfurt)

No personal data is stored. The analysis data is evaluated per user population. It is not possible to draw conclusions about individual users.

9 Software ergonomics

AppNavi and all its components are designed to be as simple and intuitive to use as possible.

In this area, various tests have been successfully passed at group works councils. The focus here is on software ergonomics topics such as::

- Usability
- Consistency
- Accessibility

During development, we follow the standards and specifications according to DIN EN ISO 9241 (parts 11, 13-16, 110, 112, 125, 129, 143, 151, 154, 161, 171, 303, 306) and WCAG 2.1 Level AA (ISO/IEC 40500:2012).

10 Boundaries and limitations

The following limits and restrictions apply to the use of AppNavi:

Tenant	
Max. Subscriptions per Tenant	100
Max. Images per Tenant	1GB
Max. Image size	3MB
Max. Authors per Tenant	100
Max. Routes per Tenant	250
Max. News per Tenant	250
Max. Applications per Tenant	100
Max. Route Collections per Tenant	50
Max. Title length	50
Max. length Company	100
Max. number of backups	5GB
Max. number of User Profiles	50.000

Subscription	
Max. Routes per Subscription	50
Max. Applications per Subscription	50
Max. News per Subscription	100
Max. Route Collections per Subscription	25
Max. Authors per Subscription	100
Max. Title length	50
Max. length of Description	100

Application	
Max. number of assigned routes	50

Routes	
Max. number of steps	30

Max. number of assigned News	50
Max. number of assigned Route Collections	25
Max. Length of Title	50
Max. Length of category	50
Max. Length of pattern	128
Max. Length of Url	128
Max. Length Custom Code	5.000
Max. Length Custom CSS	5.000
Max. Size of Image	3MB
Allowed image formats: .jpg, .jpeg, .png	3
Max. Number of published Routes	30
Max. Number of published News	50
Max. Number of published Route Collections	10

Max. Length of title per Step	100
Max. Lentht of content Inhalt per Step	1.000
Max. Size per Step	100KB
Max. Length Custom Code (OnBeforeRender) / Step	1.024
Max. Length Custom Code (OnAfterRender) / Step	1.024
Max. Number of captures per Step	3
Max. Length of Url (relative path)	512
Max. Size of route	3MB
Max. Length of Description	100

News	
Max. Length of Title	50
Max. Length of Description	100
Max. Length of Content	1.000
Max. Sie of Image	3MB
Allowed image formats: .jpg, .jpeg, .png	3

Routen Collections	
Max. Length of Title	50
Max. Length of Description	100
Max. Length of Content	1.000
Max. Size of Image	3MB
Allowed image formats: .jpg, .jpeg, .png	3
Max. Number of assigne Routes	15

User	
Max. Length of First Name	25
Max. Length of Last name	25
Max. Length of Phone Number	25
Max. Length of Email Address	50
Max. Lenth of user name	25

11 Other service areas

In addition to providing software products, AppNavi offers other services:

SW implementation

- **Consulting:** requirements gathering, adoption design, learner journeys, KPI definition, etc**.**
- **Set-Up Services:** Integration, customizing, styling, segmentations, content creation
- **Training:** Training of authors
- **Go Live Support**

SW support

- **Standard product support:** provision of releases, troubleshooting (incident management)
- **Optional support services:** 1st/2nd level for standard applications, more extensive service hours, individual service levels, additional support languages, dedicated support