



FF ADMIN

Anleitung

bis v1.3.x



Open Source Software
entwickelt durch JK Effects
von Julian Krauser
09. Februar 2025

Inhaltsverzeichnis

Inhaltsverzeichnis	II
1 Einleitung	1
2 Installation	2
2.1 Docker	2
2.1.1 Docker-Compose	2
2.2 Git	6
2.3 Konfiguration	8
2.4 Einrichtung	11
2.5 Update der Version	12
3 Konzepte	13
3.1 Stammdaten	13
3.2 Berechtigungen	13
3.3 Engines	14
3.3.1 Template-Engine	14
3.3.2 Query-Engine	14
4 Module	15
4.1 Mitgliederverwaltung & Stammdaten	15
4.2 Kalender	17
4.3 Protokolle	20
4.4 Newsletter	21
4.5 Backups	21
4.6 Query Builder & Query Store	21
4.7 Templates & Template Builder	21
4.8 Benutzer & Rollenverwaltung	21
4.9 Webapi	21
5 Ökosystem FF Admin	22
6 Roadmap	23

1 Einleitung

FF Admin - Die zentrale Verwaltungssoftware für Feuerwehren und Vereine

FF Admin ist eine vielseitige Mitgliederverwaltungssoftware, die als Herzstück eines wachsenden Ökosystems dient. Neben der Mitgliederverwaltung ermöglicht das Programm die Organisation von Terminkalendern, die Erstellung von Newslettern und Protokollen sowie - in Zukunft - die Verwaltung von Gerätschaften und Prüfplänen.

Obwohl FF Admin in erster Linie für Feuerwehren konzipiert ist, kann es dank seines modularen Aufbaus auch für andere Organisationen angepasst werden. Die frei definierbaren Stammdaten ermöglichen einen flexiblen Einsatz, so dass die Software optimal an die individuellen Bedürfnisse angepasst werden kann.

2 Installation

FF Admin kann über mehrere Wege betrieben werden. Zum einen werden Docker-Images versioniert zur Verfügung gestellt. Weiterhin kann auch das Release Projekt heruntergeladen und verwendet werden.

2.1 Docker

Disclaimer: Die Anleitung zum Betrieb von FF Admin mit Docker setzt Kenntnisse mit Docker und Docker-Compose voraus.

Die Docker-Images können gemeinsam über eine Compose-File konfiguriert und gestartet werden. Auch können die Images einzeln gestartet werden.

Die Docker-Images sind versioniert. Der `<tag>` des Images kann entweder `latest` für die neueste Version oder `vX.Y.Z` für eine bestimmte Version sein. Die Versionen können auch in den Releases der Repositories der Anwendungen nachgeschlagen werden. Dort lassen sich auch Informationen zu neuen Funktionen, Änderungen oder Fehlerbehebungen der jeweiligen Funktion finden.

2.1.1 Docker-Compose

App

```
1 ff-admin-app:
2   image: docker.registry.jk-effects.cloud/ehrenamt/ff-admin/app:<version>
3   container_name: ff_admin
4   restart: unless-stopped
5   ports:
6     - "80:80"
7   environment:
8     - SERVERADDRESS=<backend_url>
9     - APPNAMEOVERWRITE=<appname>
10    - IMPRINTLINK=<imprint link>
11    - PRIVACYLINK=<privacy link>
12    - CUSTOMLOGINMESSAGE=<betrieben von xy>
13   volumes:
14     - <volume|local path>/favicon.ico:/usr/share/nginx/html/favicon.ico
15     - <volume|local path>/favicon.png:/usr/share/nginx/html/favicon.png
16     - <volume|local path>/Logo.png:/usr/share/nginx/html/Logo.png
```



Anleitung zu FF Admin bis v1.3.x – Installation

Die Verwendung der Werte des Typs Environment werden unter dem Punkt Konfiguration ([Abschnitt 2.3](#)) erklärt.

Alle Environment Werte sind Optional und haben Standard-Werte.

Ist ein Wert optional und hat keinen Fallback, so wird in der Anwendung nichts angezeigt.

Die Volumes dienen zur erweiterten Personalisierung der App mit eigenem Logo der Feuerwehr oder des Vereins. Hiervon betroffen ist das Icon im Browser-Tab, jede Anzeige des FF Admin Logos innerhalb der App und das Icon, wenn die WebApp auf einem Gerät installiert wird.

Die Konfiguration der Volumes ist optional, falls Sie die Standard-Logos verwenden wollen.

Ein Teil der Logos haben eine Anforderung an die Auflösung:

Icon	Auflösung	Anzeigeort
favicon.ico	48x48 px	Browser-Tab Icon
favicon.png	512x512 px	WebApp Icon zur Installation
Logo.png	beliebig	Innerhalb der Anwendung

Die Dateien müssen exakt gleich geschrieben sein. Achten Sie deshalb auf Schreibfehler und Groß-/Kleinschreibung.

Anleitung zu FF Admin bis v1.3.x – Installation

Server

```
1  ff-admin-server:
2      image: docker.registry.jk-effects.cloud/ehrenamt/ff-admin/server:<version>
3      container_name: ff_member_administration_server
4      restart: unless-stopped
5      ports:
6          - "5000:5000"
7      environment:
8          - DB_TYPE=<database type>
9          - DB_HOST=<database host>
10         - DB_PORT=<database port>
11         - DB_NAME=<database name>
12         - DB_USERNAME=<database username>
13         - DB_PASSWORD=<database password>
14         - JWT_SECRET=<jwt secret>
15         - JWT_EXPIRATION=<jwt expiration>
16         - REFRESH_EXPIRATION=<refresh expiration>
17         - PWA_REFRESH_EXPIRATION=<pwa refresh expiration>
18         - MAIL_USERNAME=<mailadress|username>
19         - MAIL_PASSWORD=<mail password>
20         - MAIL_HOST=<mail server url>
21         - MAIL_PORT=<port>
22         - MAIL_SECURE=<boolean>
23         - CLUB_NAME=<club name>
24         - CLUB_WEBSITE=<club website>
25         - BACKUP_INTERVAL=<backup interval>
26         - BACKUP_COPIES=<backup parallel copies>
27         - BACKUP_AUTO_RESTORE=<boolean>
28         - USE_SECURITY_STRICT_LIMIT=<boolean>
29         - SECURITY_STRICT_LIMIT_WINDOW=<time window>
30         - SECURITY_STRICT_LIMIT_REQUEST_COUNT=<strict_request_count>
31         - USE_SECURITY_LIMIT=<boolean>
32         - SECURITY_LIMIT_WINDOW=<time window>
33         - SECURITY_LIMIT_REQUEST_COUNT=<request_count>
34         - TRUST_PROXY=<proxy config>
35      volumes:
36          - <volume|local path>:/app/files
```

Die Verwendung der Werte des Typs Environment werden unter dem Punkt Konfiguration (Abschnitt 2.3) erklärt.

Environment Werte können optional sein oder haben Standard-Werte.

Das Fehlen einer geforderten Variable oder die falsche Angabe eines Variablen-Werts verhindert das Starten der Anwendung.

Innerhalb dem Ordner, der dem Volume zugeordnet ist, werden Backups und Ausdrucke der

Anleitung zu FF Admin bis v1.3.x – Installation

geschriebenen Protokolle und Newsletter abgelegt.

Datenbank

Als Datenbank können MySQL, Postgres und SQLite verwendet werden. Postgres wird für den Produktiven Einsatz empfohlen.

Konfiguration von MySQL:

```
1 ff-db:
2   image: mariadb:<version (bsp 11.2)>
3   container_name: ff_db
4   restart: unless-stopped
5   ports:
6     - "3306:3306"
7   environment:
8     - MYSQL_DATABASE=<database name>
9     - MYSQL_USER=<username>
10    - MYSQL_PASSWORD=<user password>
11    - MYSQL_ROOT_PASSWORD=<root password>
12   volumes:
13     - <volume|local path>:/var/lib/mysql
```

YAML

MYSQL_USER und MYSQL_PASSWORD sind optional. Werden diese nicht gesetzt, kann der Server entweder mit dem Nutzer root und dem gesetzten MYSQL_ROOT_PASSWORD Zugang zur Datenbank erhalten, oder es wird im nachhinein ein Nutzerzugang erstellt, der Zugriff auf die erstellte Datenbank hat. MYSQL_DATABASE erstellt direkt eine Datenbank, die durch einen angelegten MYSQL_USER verfügbar ist.

Konfiguration von Postgres:

```
1 ff-db:
2   image: postgres:<version (bsp 16)>
3   container_name: ff_db
4   restart: unless-stopped
5   ports:
6     - "5432:5432"
7   environment:
8     - POSTGRES_DB=<database name>
9     - POSTGRES_USER=<username>
10    - POSTGRES_PASSWORD=<user password>
11   volumes:
12     - <volume|local path>:/var/lib/postgresql/data
```

YAML

POSTGRES_DB erstellt direkt eine Datenbank, die durch einen angelegten POSTGRES_USER verfügbar ist.

Hinweis Wenn eine Docker-Compose Datei verwendet wird, kann zusätzliche ein Netzwerk ange-

legt werden. Dadurch ist das Veröffentlichen der Datenbank-Port-Exposes nicht mehr notwendig. Das Entfernen der port-Exposes verhindert den direkten Zugriff auf die Ports von außerhalb. Ergänzt muss hierfür das `network` und die Teilhabe des Backend-Containers am Netzwerk:

1. Ergänzung zu Server und Datenbank:

```
1 networks:
2   - ff_internal
```

2. Ergänzung zur finalen Compose:

```
1 networks:
2   ff_internal:
```

3. Optionale Ergänzung zum Server:

```
1 depends_on:
2   - ff-db
```

Hierdurch kann der Server nicht starten, wenn die verwendete Datenbank nicht läuft.

2.2 Git

Eine Alternative zu Docker ist die direkte Ausführung der Anwendungen auf dem Server oder Desktop Gerät.

Hierzu müssen die App und der Server als Quellcode auf das System geladen und dort direkt verwendet werden.

Die Veröffentlichung der App und des Servers, damit diese aus dem Internet erreichbar sind, muss gesondert eingerichtet werden.

Das System muss NodeJs und die bevorzugte Datenbank installiert haben.

Für das Hosting von statischen Inhalten kann Apache oder Nginx verwendet werden. Eine Konfiguration für Nginx ist im Repo der App enthalten.

Die NodeJs Prozesse können auch durch Tools wie pm2 verwaltet werden.

Um die Konfiguration mittels ENV-Variablen an die Anwendungen weitergeben zu können, müssen `.env` Dateien erstellt werden. Hierzu kann die `.env.example` Datei kopiert und die definierten Werte ausgefüllt werden. Nicht benötigte Einträge sollten entfernt werden.

Die `env`-Datei im Frontend muss vor dem build-Prozess erstellt sein, da dort die Werte fest in den Code übernommen werden. Weiterhin muss die Datei im Frontend `.env.production` heißen. Die bestehende Datei kann modifiziert werden. Bei einer Änderung muss die App neu gebaut werden.

Anleitung zu FF Admin bis v1.3.x – Installation

Die env-Datei im Backend muss vor der Ausführung von `npm run start` angelegt sein. Bei einer Änderung der Einträge muss der Server lediglich neu gestartet werden.

Anleitung zu FF Admin bis v1.3.x – Installation

App

```
1 git clone https://forgejo.jk-effects.cloud/Ehrenamt/ff-admin.git
2 cd ff-admin
3 npm install
4 npm run build
```

Shell

Der durch `npm run build` erstellte `dist` Ordner kann mit Apache oder Nginx zur Verfügung gestellt werden.

Server

```
1 git clone https://forgejo.jk-effects.cloud/Ehrenamt/ff-admin-server.git
2 cd ff-admin-server
3 npm install
4 npm run build
5 npm run start
```

Shell









2.3 Konfiguration



Folgende Werte können zu einem Container konfiguriert werden:

Variable	Zweck	Fallback	optional
App-Variablen			
SERVERADDRESS	URL, über welche das Backend erreicht werden kann. Die URL muss mit <code>http://</code> oder <code>https://</code> starten und darf keinen Pfad beinhalten. Wenn das Backend auf der gleichen URL wie die App läuft, kann diese Variable weggelassen werden.		✓
APPNAMEOVERWRITE	Anzeige eines anderen Namens als FF Admin.	FF Admin	✓
IMPRINTLINK	Link zum Impressum des Betreibers.		✓
PRIVACYLINK	Link zur Datenschutzerklärung des Betreibers.		✓
CUSTOMLOGINMESSAGE	Nachricht auf der Login-Seite. (Bsp.: betrieben von xy)		✓
Server-Variablen			

Variable	Zweck	Fallback	optional
DB_TYPE	Folgende Datenbanktypen sind verfügbar: mysql, sqlite, postgres	mysql	✓
DB_HOST	URL zur Datenbank oder Dateipfad zur SQLite-Datenbank		✗
DB_PORT	Port der Datenbank	3306	⚠
DB_NAME	Name der Datenbank in welcher die Tabellen erstellt werden.		⚠
DB_USERNAME	Nutzername für Zugang zu Datenbank		⚠
DB_PASSWORD	Passwort zum Zugang zur Datenbank		⚠
JWT_SECRET	Zufällige Zeichenkette zur Validierung der Session-Tokens.		✗
JWT_EXPIRATION	Gültigkeitsdauer eines Session-Tokens. Format: [0-9]*(y d h m s)	15m	✓
REFRESH_EXPIRATION	Gültigkeitsdauer eines Logins nach letzter Nutzung der App im Browser Format: [0-9]*(y d h m s)	1d	✓
PWA_REFRESH_EXPIRATION	Gültigkeitsdauer eines Logins nach letzter Nutzung der installierten App Format: [0-9]*(y d h m s)	5d	✓
MAIL_USERNAME	Nutzername oder Mailadresse		✗
MAIL_PASSWORD	Passwort zum Nutzernamen oder der Mailadresse		✗
MAIL_HOST	URL des Mailservers		✗
MAIL_PORT	Port des Mailservers für Versand (SMTP). Ports sind 25, 465, 587	587	✓
MAIL_SECURE	Soll eine Secure Verbindung aufgebaut werden. Muss true sein bei Port 465.	false	✓
CLUB_NAME	Wird für TOTP Titel und Kalender-ICS verwendet.	FF Admin	✓
CLUB_WEBSITE	Wird für Kalender-ICS verwendet		✓
BACKUP_INTERVAL	Wie viele Tage Abstand sollen zwischen Backups liegen? (min. 1)	1	✓
BACKUP_COPIES	Wie viele parallele Kopien von Backups sollen parallel Verfügbar sein? (min. 1)	7	✓

Variable	Zweck	Fallback	optional
BACKUP_AUTO_RESTORE	Soll das neueste Backup bei Server-Start automatisch geladen werden, wenn die Datenbank als leer erkannt wird?	true	<input checked="" type="checkbox"/>
USE_SECURITY_STRICT_LIMIT	Soll ein Anfrage-Limit für Login, Reset und Co gesetzt werden? In diesem Fall ist der Nutzer nicht angemeldet, sondern versucht es.	true	<input checked="" type="checkbox"/>
SECURITY_STRICT_LIMIT_WINDOW	Über welches Zeitfenster soll das Limit angewandt werden? Format: [0-9]*(y d h m s)	15m	<input checked="" type="checkbox"/>
SECURITY_STRICT_LIMIT_REQUEST_COUNT	Wie viele fehlerhafte Anfragen müssen gesendet werden, bis das Limit aktiviert ist?	15	<input checked="" type="checkbox"/>
USE_SECURITY_LIMIT	Soll ein Anfrage-Limit für Anfragen innerhalb der App gesetzt werden? In diesem Fall ist der Nutzer angemeldet.	true	<input checked="" type="checkbox"/>
SECURITY_LIMIT_WINDOW	Über welches Zeitfenster soll das Limit angewandt werden? Format: [0-9]*(y d h m s)	1m	<input checked="" type="checkbox"/>
SECURITY_LIMIT_REQUEST_COUNT	Wie viele fehlerhafte Anfragen müssen gesendet werden, bis das Limit aktiviert ist?	500	<input checked="" type="checkbox"/>
TRUST_PROXY	Wird der Server hinter einem Proxy betrieben und Rate-Limit verwendet? Ist dieser Wert nicht gesetzt, wird davon ausgegangen, dass kein Proxy verwendet wird. Folgende Werte können gesetzt werden: true / false Anzahl der Proxies: [0-9]* IP-Adresse des Proxy: ip IP-Adressen der Proxy: ip1,ip2,...		<input checked="" type="checkbox"/>

Variable	Zweck	Fallback	optional
 Database-Variablen			
MYSQL_DATABASE	Name der Datenbank, die bei Erstellung direkt angelegt wird.		
MYSQL_USER	Benutzername des Users, der bei Erstellung direkt angelegt wird.		
MYSQL_PASSWORD	Passwort zum User, das bei Erstellung gesetzt wird.		
MYSQL_ROOT_PASSWORD	Passwort für den User root, das bei Erstellung gesetzt wird.		
POSTGRES_DB	Name der Datenbank, die bei Erstellung direkt angelegt wird.		
POSTGRES_USER	Benutzername des Users, der bei Erstellung direkt angelegt wird.		
POSTGRES_PASSWORD	Passwort zum User, das bei Erstellung gesetzt wird.		

: Ein Fehlen dieser Variable verhindert das Starten der Anwendung! : Bei Verwendung von SQLite sind diese Variablen nicht notwendig!

Hinweis: Eine fehlerhafte Konfiguration der optionalen oder geforderten Variable verhindert das Starten der Anwendung.

Hinweis: Eine Änderung der Datenbank übernimmt die Daten nur automatisch in die neue Datenbank, wenn BACKUP_AUTO_RESTORE aktiviert ist und ein Backup angelegt ist. Es werden dann die Daten des gefundenen Backups in die neue Datenbank eingefügt.

2.4 Einrichtung

Um die Anwendung nutzen zu können, kann ein erster Administrator-Account wie folgt erstellt werden:

1. **Admin Benutzer erstellen:** Erstellen Sie einen Admin Benutzer unter dem Pfad /setup, um auf die Mitgliederverwaltung Zugriff zu erhalten. Nach der Erstellung des ersten Benutzers wird der Pfad automatisch geblockt.
2. **Rollen und Berechtigungen:** Unter Benutzer > Rollen können die Rollen und Berechtigungen für die Benutzer erstellt und angepasst werden.
3. **Nutzer einladen:** Unter Benutzer > Benutzer können weitere Nutzer eingeladen werden. Diese erhalten dann eine E-Mail mit einem Link, um ein TOTP zu erhalten.

2.5 Update der Version

Um eine Version auf eine Neuere zu aktualisieren, muss meist nur der Docker-Tag oder das Repo ersetzt werden.

Wer Docker mit `latest` nutzt, kann das neue Image direkt mit `docker pull` neu beziehen und dann den Container neustarten.

Informationen zu neuen Versionen können innerhalb der App im Account des Eigentümers oder in den Release-Pages gefunden werden.

Die Releases beinhalten Informationen zu einem Update und was zu beachten ist. So enthalten die Release-Informationen beispielsweise Vorbereitungen vor einem Update.

Bei Verwendung mittels Git, müssen die Repos neu bezogen werden. Anschließend müssen die Dependencies neu installiert und die Anwendungen neu gebaut werden.

3 Konzepte

FF Admin basiert auf mehreren Konzepten, die das System modular und flexibel machen, so dass es an verschiedene Anwendungsfälle angepasst werden kann.

3.1 Stammdaten

Stammdaten sind grundlegende Basisdaten, die als Grundlage für weitere Einträge dienen. Dazu gehören z.B. Anrede, Mitgliedsstatus oder verschiedene Arten von Kalendereinträgen.

Diese Daten sind frei definierbar, so dass die Benutzer ihre eigene Namensgebung festlegen und sicherstellen können, dass alle benötigten Einträge zur Verfügung stehen.

Damit Einträge angelegt oder verwendet werden können, müssen **zuvor** die entsprechenden Stammdaten angelegt worden sein.

3.2 Berechtigungen

Das Berechtigungssystem ist tief in FF Admin integriert und steuert den Zugriff von Benutzern oder API-Clients auf verschiedene Sektionen und Module. Berechtigungen bestimmen, ob jemand Daten lesen, erstellen, aktualisieren oder löschen kann.

Berechtigungen werden immer summiert:

Ein Benutzer erhält alle Berechtigungen, die ihm direkt oder indirekt über Rollen zugewiesen wurden.

Der Berechtigungseeditor ist in Sektionen und Module unterteilt. Wenn ein Benutzer eine Berechtigung für einen Abschnitt erhält, gilt diese automatisch auch für alle untergeordneten Module.

Hinweis: Um Einträge in einem Modul, das auf Stammdaten zugreift, anlegen oder bearbeiten zu können, erhält der Benutzer automatisch Leserechte auf die Stammdaten. Dies bedeutet jedoch nicht, dass der Benutzer die Stammdaten direkt in der Anwendung sehen kann - hierfür benötigt er eine explizite Berechtigung für den entsprechenden Abschnitt oder das entsprechende Modul.

3.3 Engines

FF Admin stellt eine Reihe von systemweiten Funktionen zur Verfügung, die vom Benutzer konfiguriert und in verschiedenen Modulen verwendet werden können.

3.3.1 Template-Engine

Die Template-Engine ermöglicht es dem Benutzer, eigene Formatvorlagen zu erstellen und Standardvorlagen damit zu ersetzen. Ein Modul kann Vorlagen für Kopf- und Fußzeilen sowie für den Hauptteil eines Dokuments verwenden.

Diese Vorlagen werden zum Drucken von PDFs oder zum Versenden von E-Mails verwendet. Sie werden mit Daten aus FF Admin gefüllt und anschließend exportiert oder versendet.

Eine detaillierte Anleitung zur Verwendung finden Sie unter **Module -> Templates & Template Builder** ([Abschnitt 4.6](#)).

3.3.2 Query-Engine

Die Query-Engine ermöglicht das Erstellen und Ausführen von individuellen Abfragen auf die Mitgliederdaten. Diese Abfragen können gespeichert und in anderen Modulen wieder verwendet werden.

Beispiele von Abfragen wären:

- Mitglieder eines bestimmten Mitgliedschaftstyps
- Mitglieder einer bestimmten Mitgliedschaftsdauer
- Mitglieder eines bestimmten Alters
- ...

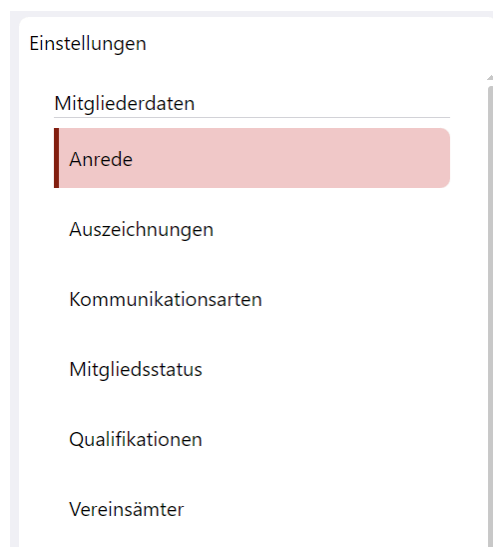
Eine detaillierte Anleitung zur Benutzung finden Sie unter **Module -> Query Builder & Query Store** ([Abschnitt 4.7](#)).

Hinweis: Die Query-Engine ermöglicht nur das Lesen von Mitglieder- und mitgliederbezogenen Daten. Das Schreiben oder Verändern von Daten ist nicht möglich.

4 Module

FF Admin nutzt verschiedenste Module, um Daten zu organisieren, die Verwaltung zu vereinfachen und Berechtigungen sinnvoll setzen zu können.

4.1 Mitgliederverwaltung & Stammdaten





Die Mitgliederverwaltung ermöglicht die strukturierte Erfassung und Verwaltung von Mitglieder Daten. Bevor einzelne Mitglieder Datensätze angelegt werden können, müssen zunächst die Mitgliederstammdaten definiert werden. Diese Stammdaten dienen als Grundlage für die spätere Datenerfassung und ermöglichen eine einheitliche, standardisierte Verwaltung. Sind die Stammdaten einmal hinterlegt, können sie bei der Erfassung neuer Mitglieder verwendet werden.

Dazu gehören grundlegende Kategorien wie Anrede, Auszeichnungen, Kommunikationskanäle, Mitgliedsstatus, Qualifikationen und Vereinsämter.

Alle Stammdaten können frei ausgefüllt werden und einem Mitglied je nach Typ mehrfach zugewiesen werden. Dabei können diese unterschiedliche Inhalte speichern. Ein Titel oder eine Bezeichnung können dabei nicht doppelt vorkommen.

- **Anrede:** Anreden können je nach Bedarf mit Werten wie Herr, Frau, Dr. oder Prof. erstellt werden.
- **Auszeichnungen:** Titel enthalten nur den Titel.
- **Kommunikationskanäle:** Kommunikationskanäle enthalten sowohl die Bezeichnung als auch die dem Kanal zugeordneten Kommunikationsarten. Folgende Felder können zugeordnet werden: Telefonnummer, E-Mail-Adresse, Postleitzahl, Stadt/Ort, Straße, Hausnummer und Hausnummernzusatz.
- **Mitgliedsstatus:** Mitgliedsstatus sind frei wählbar und können die Zugehörigkeit zu einem Verein oder einer Vereinsgruppe widerspiegeln. Beispiele wären „aktiv“, „passiv“, „fördernd“ oder „Tischtennis“ oder „Fußball“.
- **Qualifikationen:** Qualifikationen können frei vergeben werden. Zusätzlich kann eine Beschreibung angegeben werden. Beispiele sind „Atemschutzgeräteträger“, „Gruppenführer“...
- **Vereinsämter:** Vereinsämter können ebenfalls frei belegt werden. Dabei können alle Funktionen im Verein von „Vorstand“ bis „EDV“ angelegt werden.

Krauser, Julian

Übersicht

MitgliedschaftKommunikationAuszeichnungenQualifikationenVereinsämter

Interne Id

Anrede

sir

Vorname

Julian

Nachname

Krauser

Nameaffix

Geburtsdatum

20.09.2003

Statistiken zur Mitgliedschaft

aktiv für gesamt 2644 Tage ~> 7 years 2 months 4 days

Erster Eintrag Mitgliedschaft

2017-11-13 bis heute: aktiv

Innerhalb der Mitgliederansicht können dann individuelle Mitgliederprofile angelegt und gepflegt werden. Diese enthalten persönliche Daten wie Name, Geburtsdatum sowie spezifische Informationen zur Mitgliedschaft wie Eintrittsdatum, Qualifikationen oder verliehene Auszeichnungen. Darüber hinaus bietet das System eine automatische Berechnung der Mitgliedschaftsdauer je zugewiesenem Status, um die aktive Zeit eines Mitglieds jederzeit nachvollziehen zu können.

Die Einträge zu einem Mitglied umfassen dann jeweils den Basisdatensatz und weitere Details, wie z.B. einen Zeitraum, in dem dieser Eintrag gültig ist oder war. Einige Eingabemöglichkeiten weisen Besonderheiten auf.

Mitgliedschaften werden als Zeiträume eines bestimmten Status angegeben. Durch die Angabe eines neuen Zeitraums wird ein bisher aktiver Zeitraum beendet und das Enddatum auf den Tag vor Beginn des neuen Zeitraums gesetzt. Derzeit wird nicht geprüft, ob sich Zeiträume überschneiden. Zusätzlich kann bei einem Eintrag ein Grund für das Ende des Eintrags angegeben werden. Beispiele hierfür wären der Austritt oder der Tod eines Mitglieds.

Bei einem **Kommunikationskanal** können nur die Felder ausgefüllt werden, die in den Stammdaten hinterlegt wurden. Zusätzlich kann angegeben werden, welche der Adressen bevorzugt verwendet werden soll. Dadurch ist es möglich, mehrere Telefonnummern oder Adressen zu

erfassen, ohne die Übersicht zu verlieren. Weiterhin kann angegeben werden, ob ein Kommunikationskanal für den Versand eines Newsletters verwendet werden soll. Hierbei ist zu beachten, dass ein Newsletter derzeit nicht an eine Telefonnummer versendet werden kann.

Im Feuerwehrkontext kann bei einer Telefonnummer angegeben werden, ob diese für eine SMS-Alarmierung verwendet werden kann. Der Versand einer SMS an Mitglieder im Alarmfall ist nicht in FF Admin integriert und muss extern konfiguriert werden. Die Angabe erleichtert jedoch die Identifikation von Telefonnummern, die für diesen Anwendungsfall verwendet werden können.

Auszeichnungen können einem Mitglied mit einem Übergabedatum zugeordnet werden. Dabei kann auch vermerkt werden, ob die Annahme vom Mitglied verweigert oder vom Verein verweigert wurde. Optional kann eine Notiz hinzugefügt werden. Die Notiz kann die Gründe für die Nichtannahme oder eine verspätete Übergabe enthalten.

Qualifikationen können auch mit einem Zeitraum angelegt werden. Ein Eintrag kann weiterhin eine Notiz enthalten, z.B. eine Einschränkung der Qualifikation. Außerdem kann angegeben werden, warum die Qualifikation zurückgezogen wurde oder nicht mehr gültig ist.

Vereinsämter können ebenfalls mit den Zeiträumen der Ausübung angelegt werden. Auch hier besteht die Möglichkeit, eine Notiz hinzuzufügen.

Anmerkung: In jeder erfassbaren Datenkategorie können Einträge mehrfach vorkommen. Eine Überprüfung auf sich überschneidende Zeiträume oder Doppelerfassungen muss derzeit durch den Anwender erfolgen.

4.2 Kalender

Der FF Admin beinhaltet ein Modul für die Verwaltung von Terminen. Diese Termine können den Mitgliedern oder anderen Interessierten über die integrierte öffentliche Ansicht oder direkt als WebCal-Link zur Verfügung gestellt werden.



Auch im Kalendermodul müssen Stammdaten angelegt werden.

Die Stammdaten dienen dazu, die Kalendereinträge bestimmten Kategorien zuzuordnen. Die Kategorien enthalten eine Farbe und können dauerhaft öffentlich oder durch ein Passwort geschützt sein. Der Schutz einer Kategorie mit einem Passwort erschwert das automatisierte Testen und Auslesen von Terminen, da dann neben dem Namen der Kategorie auch das Passwort im Link angegeben werden muss.

Kalender

Monat

Woche

Februar 2025

<

Heute

>

Mo	Di	Mi	Do	Fr	Sa	So
KW5 27	28	29	30	31 19:30 Winterschulung	1	2
KW6 3 19:30 Winterschulung	4	5	6	7 19: Atemschutzgeräte	8	9
KW7 10	11	12	13	14 18 Uhr Aufbau Feuerwehr	15 20 Uhr Feuerwehrball	16 + weitere 2
KW8 17	18	19	20	21	22	23
KW9 24	25	26	27	28	1	2
KW10 3	4	5	6	7	8	9

Termin erstellen

Terminotyp

FW-Übung

Titel

Beschreibung (optional)

☐ ganztägig

Startzeit

06.02.2025 01:00

Endzeit

07.02.2025 01:00

Ort (optional)

erstellen

abbrechen

Der Kalender ermöglicht die direkte Verwaltung von Terminen. Durch die Auswahl von Tagen in der Monatsansicht oder Zeiträumen in der Wochenansicht können Zeitbereiche direkt in das Formular zur Erstellung eines Termins übernommen werden.

Der vorgewählte Zeitraum kann im erscheinenden Formular jederzeit geändert werden. Bei ganztägigen Terminen können dann nur noch direkte Datumsangaben und bei zeitbegrenzten Terminen Daten mit Uhrzeit ausgewählt werden.

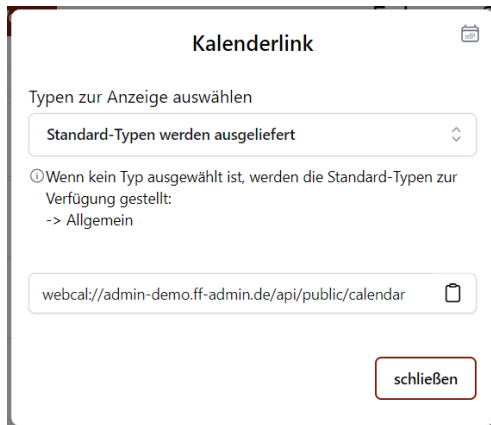
Weiterhin besteht ein Termin aus einem Terminotyp bzw. einer Kategorie, einem Titel, einer Beschreibung und einem Ort.

Die Textfelder sind frei wählbar und werden nicht auf Duplikate oder Korrektheit geprüft.

Alle Daten eines Termins sind analog zu diesem Formular in der öffentlichen Kalenderansicht sichtbar. Die Daten werden auch vollständig über WebCal übertragen.

Die öffentliche Ansicht des Kalenders ist in FF Admin unter dem URL-Pfad `/public/calendar` verfügbar und zeigt alle Termine der Kategorien an, die mit Standard Kalender Auslieferung erstellt wurden. Die Ansicht kann als angemeldeter Nutzer über das Kalender-Icon im Kalender-Link-Dialog erreicht werden.

Die Kategorien können weiterhin verwendet werden, um spezielle Links anzulegen, die Termine nur bestimmten Mitgliedern zur Verfügung stellen.



Die Freigabe des Kalenders mit einem WebCal-Link kann über eine eigene Dialogbox verwaltet werden. Die Dialogbox kann über das Link-Icon in der rechten oberen Ecke der Kalenderansicht geöffnet werden.

Hier kann ausgewählt werden, welche Kategorien über einen Link verfügbar sein sollen. Die Kategorien werden - bei Bedarf mit der Passphrase - automatisch in den Link geschrieben. Sind Kategorien ausgewählt, kann zusätzlich die Standard-Auslieferung aktiviert werden.

Ansonsten werden nur Termine der ausgewählten Kategorien ausgeliefert. Dieser kann dann kopiert und direkt in einen Kalender wie der von Google oder von Apple eingefügt werden.

4.3 Protokolle

Start Mitgliederverwaltung, 2024-11-27

Übersicht Anwesenheit Abstimmungen Beschlüsse Protokoll Druck

Titel
Start Mitgliederverwaltung

Datum
27.11.2024

Startzeit
16:00:00

Endzeit
18:00:00

Zusammenfassung

Normal Sans Serif B I U S ” </> ↺ ↻ ↺ ↻ ↺ ↻ ↺ ↻

Veröffentlichung der Demo der Mitgliederverwaltung

FF Admin bietet auch die Möglichkeit, Protokolle zu erfassen. Die Protokolle können An- und Abwesenheiten, Abstimmungen, Beschlüsse sowie die besprochenen Themen enthalten. Die Protokolle können auch als PDF exportiert werden.

Jedes Protokoll enthält einen Titel sowie ein Datum mit Anfangs- und Endzeit. Zusätzlich kann eine Zusammenfassung verfasst werden, die beim Ausdruck des Protokolls an den Anfang gestellt wird.

Die **Anwesenheit** kann nur Mitglieder erfassen, die in der Mitgliederverwaltung angelegt wurden. Bei den Anwesenheiten können Mitglieder gesucht und dem Protokoll hinzugefügt werden. Zusätzlich kann gespeichert werden, ob ein Mitglied entschuldigt oder unentschuldigt abwesend war.

Je nach Anwesenheit werden die Mitglieder auf dem pdf unter verschiedenen Positionen ausgedruckt.

Abstimmungen können ebenfalls erfasst werden. Eine Abstimmung besteht aus einem Titel und einer Beschreibung. Zusätzlich kann das Ergebnis mit der Anzahl der Ja-, Nein- und Enthaltungsstimmen angegeben werden.

Die **Beschlüsse** sind als separate Auflistung von Sitzungen gedacht. Diese sollen noch einmal klar und gesondert hervorheben, was alles beschlossen wurde.

Das **Protokoll** selbst besteht aus vielen Punkten, die in der Sitzung besprochen wurden. Es wird immer mit Titel und Beschreibung erfasst.

Jedes Protokoll kann gedruckt werden. Die erstellten pdf-Dateien können heruntergeladen oder direkt in FF Admin angezeigt werden. Jedes pdf enthält die Daten des Protokolls und ordnet diese in Listen an.

Hinweis: Die Wolke in der rechten oberen Ecke zeigt den Status der Synchronisation an. Durch Klicken auf eine Wolke mit einem Pfeil kann die Synchronisation manuell durchgeführt werden. Ansonsten wird alle 10 Sekunden synchronisiert.

Hinweis: Ein Protokoll kann derzeit nicht kollaborativ geschrieben werden. Bei paralleler Arbeit am Protokoll überschreiben sich die Status der Bearbeiter immer gegenseitig.

4.4 Newsletter

4.5 Backups

4.6 Query Builder & Query Store

4.7 Templates & Template Builder

4.8 Benutzer & Rollenverwaltung

4.9 Webapi

5 Ökosystem FF Admin

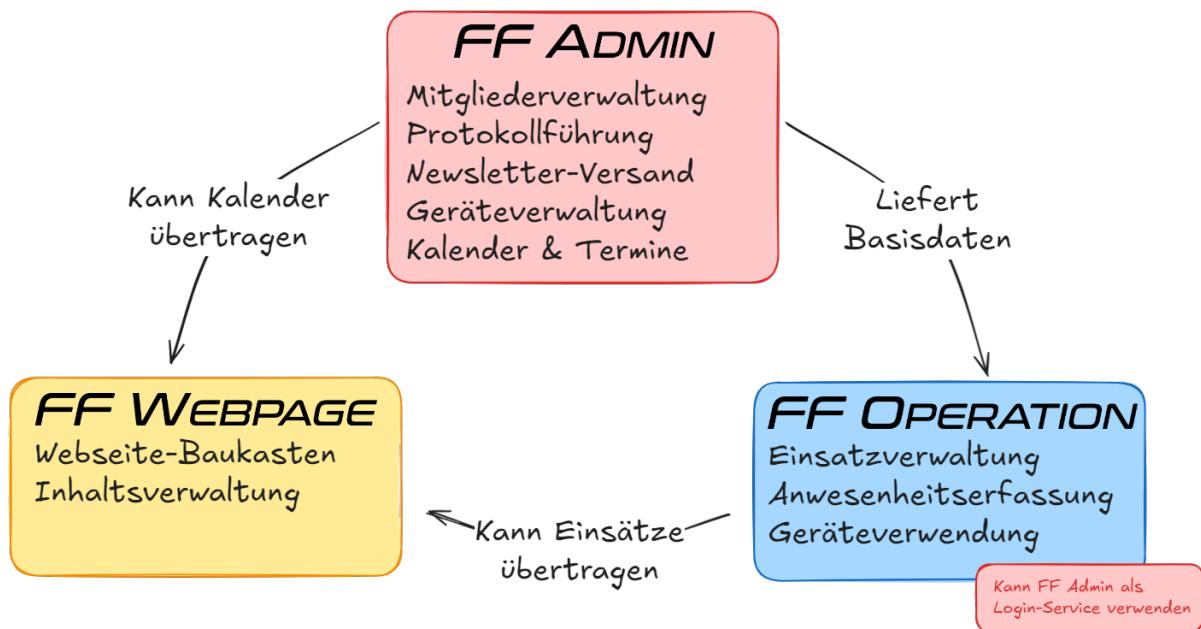


Abbildung 8: Übersicht zum Ökosystem

Das **FF Ökosystem** bietet eine zentrale Lösung für die Verwaltung von Mitgliedern, Einsätzen und Inhalten.

FF Admin ist das Herzstück und dient neben der Mitglieder- und Geräteverwaltung auch als zentraler Login-Service für alle Module. Es stellt Stammdaten bereit und ermöglicht mit allen seinen Modulen die effiziente Verwaltung einer Feuerwehr oder Vereins.

FF-Webpage erleichtert die Erstellung und Verwaltung von Webseiteninhalten. Sie kann Kalenderdaten aus FF Admin übernehmen und Einsatzinformationen aus FF Operation veröffentlichen, um Mitglieder und Öffentlichkeit auf dem Laufenden zu halten.

FF Operation unterstützt das Einsatzmanagement, die Anwesenheitserfassung und den Geräteeinsatz. Es nutzt die Mitgliederdaten, Gerätschaften und co aus FF Admin und kann erfasste Einsätze an die Webseite weiterleiten.

Disclaimer:

FF Operation befindet sich derzeit in Entwicklung und wurde noch nicht veröffentlicht.

6 Roadmap

Folgende Funktionalitäten sind für FF Admin in Planung (Auszug):

- **Calendar Link Dictionary:** Speicherung von Kalender-Link-Configs mit Aliase wie eine Name.
- **Reihenfolge von Protokoll-Inhalten:** Änderung der Reihenfolge von Abstimmungen, Beschlüssen und TOPs.
- **Mitglieder Ausdruck:** Druck der Daten eines Mitglieds anhand eines Templates.
- **Listen Ausdruck:** Druck von Listen mit Daten eines Queries anhand eines Templates.
- **Verbesserung der Template-Erstellung:** Verbesserung oder Änderung des Prozesses und Interfaces zu Erstellung eigener Templates.
- **Query Builder Erweiterung:** Erweiterung der Abfrage-Funktionalitäten des Query Builders im Bereich der Sortierung und Daten-Verbindung.
- **Kalendereinträge und Webpage:** Versand der Eingetragenen Termine des Kalenders an die Webseite mit der Möglichkeit von Änderungen eines Termins.
- **Geräteverwaltung & Prüfpläne:** Erfassung von Gerätschaften mittels Barcode und Erstellung von Prüfplänen.
- **Erinnerungen:** Versand von Erinnerungen zu anstehenden Prüfungen oder Wartungen.
- **Finanzen I:** Speicherung und Erstellung von Rechnungen als pdf mit XML-Einbettungen für E-Rechnungen.
- **Finanzen II:** Darstellung der Einnahmen und Ausgaben in einem Graphen.