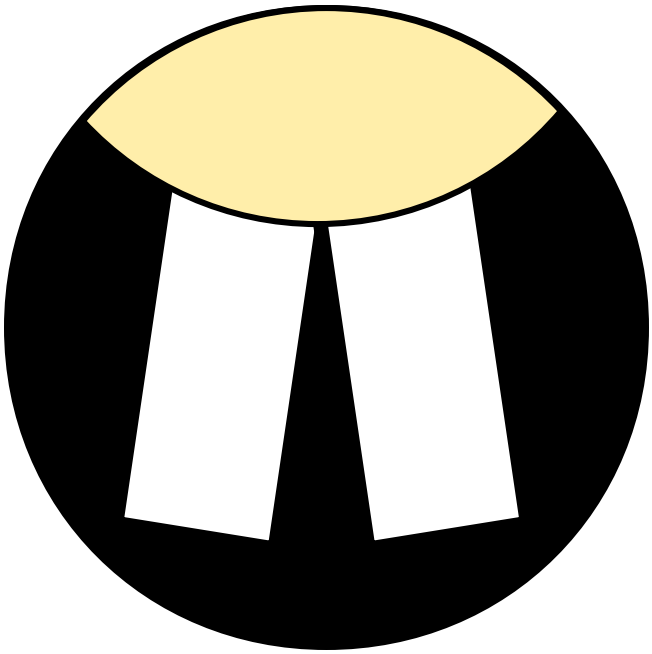

Pfarrplaner Administratorhandbuch

Installation, Betrieb, Updates und Wartung

Christoph Fischer

Pfarrplaner v.2026.12.0, Stand: 27.05.2026

Inhaltsverzeichnis

Administratorhandbuch

Dieses Handbuch beschreibt Pfarrplaner aus Sicht von Administratorinnen und Administratoren. Es erklärt Installation, Updates, Betrieb, Sicherheit und Wartung.

Schnellzugriff

Dieses Handbuch ist richtig für Sie, wenn ...

- Sie eine neue Instanz einrichten.
- Sie Updates einspielen oder Releases vorbereiten.
- Sie Mail, Cron, Queue, Speicher, Backups und Sicherheit betreuen.
- Sie Benutzerrechte und administratorische Einstellungen verstehen möchten.

Andere Handbücher

Für die tägliche Arbeit in der Oberfläche gibt es das Benutzerhandbuch.

Für Architektur, Entwicklung und API gibt es das Technische Handbuch.

1. Einführung

Pfarrplaner braucht im laufenden Betrieb mehr als nur funktionierenden Programmcode. Eine produktive Instanz braucht eine passende Serverumgebung, regelmäßige Updates, sichere Zugangsdaten, laufende Sicherungen und einen klaren Blick auf Protokolle, Hintergrundaufgaben und Speicher.

Dieses Handbuch ist deshalb kein Entwicklerhandbuch. Es beschreibt die praktische Verantwortung einer Person oder eines Teams, das Pfarrplaner für eine Gemeinde, einen Kirchenbezirk oder eine größere Trägerstruktur bereitstellt.

1.1. Typische Aufgaben der Administration

- Server und Laufzeitumgebung bereitstellen.
- Die Anwendung mit Datenbank, Mail und Dateispeicher verbinden.
- Benutzerkonten mit den richtigen Rollen anlegen.
- Updates geordnet und mit Rückfallmöglichkeit einspielen.
- Backups, Wiederherstellung und Protokolle im Blick behalten.
- Externe Integrationen, Dateiausgabe und Browser-Automatisierung funktionsfähig halten.

1.2. Wichtige Grundentscheidung

Vor der ersten Installation sollten Sie festlegen, welchen Betriebsweg Sie dauerhaft nutzen möchten:

1. **Klassische Installation**: Quellcode mit `git clone`, Abhängigkeiten mit Composer und npm, Einrichtung mit dem neuen CLI-Installer `php artisan pfarrplaner:install`.
2. **Docker-Compose-Installation**: Container für App, Datenbank und ergänzende Dienste, danach ebenfalls der CLI-Installer innerhalb der Containerumgebung.

Beide Wege sind möglich. Wichtig ist vor allem, dass Sie einen Weg sauber dokumentieren und danach bei Updates dabei bleiben.

1.3. Was dieses Handbuch abdeckt

- Voraussetzungen für Server, Laufzeit und Zusatzprogramme
- Einrichtungsabläufe für beide Installationswege
- Updatepfade für beide Installationswege
- Betrieb, Wartung und Sicherheitsaufgaben
- typische Fehlerbilder

Details zur internen Architektur, zu Quellcode-Strukturen und zur API stehen im Technischen Handbuch.

2. Systemvoraussetzungen

Pfarrplaner ist eine Laravel-Anwendung mit Vue-Frontend und zusätzlichen Werkzeugen für Dateiausgaben, Browser-Automatisierung und Dokumenterzeugung. Für einen stabilen Betrieb müssen deshalb mehrere Laufzeiten zusammenpassen.

2.1. Mindestbestandteile einer produktiven Umgebung

- Linux-Server mit dauerhaftem Speicher
- Webserver wie Apache oder Nginx
- PHP 8.4
- Composer
- Node.js mit npm für Build und manche Updates
- MySQL, MariaDB, PostgreSQL, SQL Server oder SQLite
- Cron
- Prozessüberwachung für Queue-Worker, falls Warteschlangen genutzt werden

2.2. PHP-Anforderungen

Aus `composer.json` ergeben sich mindestens diese harten Anforderungen:

- PHP `^8.4`
- PHP-Erweiterung `ext-dom`
- PHP-Erweiterung `ext-zip`

Zusätzlich braucht Laravel in der Praxis weitere übliche Erweiterungen, zum Beispiel:

- `mbstring`
- `openssl`
- `pdo`
- zum gewählten Datenbanksystem passende PDO-Erweiterung
- `fileinfo`
- `tokenizer`
- `xml`
- `ctype`
- `json`

Wenn Sie Dateiausgaben, Importe oder Office-Dokumente nutzen, sollten Sie fehlende Basis-Erweiterungen vor der Installation gezielt prüfen, nicht erst bei der ersten Fehlermeldung.

2.3. Datenbank

Der neue Installer unterstützt:

- `mysql`
- `sqlite`
- `pgsql`
- `sqlsrv`

Für produktive Mehrbenutzer-Instanzen empfiehlt sich in der Regel MySQL oder MariaDB. SQLite ist vor allem für kleine Test- oder Demo-Umgebungen praktisch.

2.4. Node.js, npm und Frontend-Build

Pfarrplaner nutzt Vite und ein modernes npm-Ökosystem. Für produktive Builds brauchen Sie:

- eine aktuelle Node.js-LTS-Version
- `npm`
- ausreichend Arbeitsspeicher für `npm install` und `npm run build`

Node wird nicht nur bei der Erstinstallation gebraucht. Auch Updates können neue npm-Abhängigkeiten oder neue Assets mitbringen.

2.5. Chromium, Puppeteer und Browsershot

Pfarrplaner verwendet unter anderem:

- `puppeteer`
- `spatie/browsershot`

Darum muss auf dem Zielsystem ein lauffähiger Chromium- beziehungsweise Chrome-Stack vorhanden sein. Prüfen Sie besonders:

- Chromium oder Google Chrome ist installiert.
- Alle nötigen Systembibliotheken für Headless-Chromium sind vorhanden.
- Der Benutzer der Webanwendung darf den Browser im Headless-Betrieb starten.
- Temporäre Verzeichnisse und Cache-Verzeichnisse sind schreibbar.

Fehlt diese Laufzeit, schlagen PDF-, Screenshot- oder Browser-basierte Ausgaben oft erst später im Betrieb fehl.

2.6. Webserver

Der Webserver muss auf das Verzeichnis `public/` zeigen. Alles andere bleibt außerhalb des direkt erreichbaren Document Roots.

Wichtig sind außerdem:

- HTTPS
- ausreichend Upload-Größen
- saubere Weiterleitung auf die öffentliche Basis-URL
- Schutz gegen direkten Zugriff auf interne Dateien außerhalb von `public/`

2.7. Mail

Pfarrplaner verschickt Systemmails. Deshalb brauchen Sie:

- gültige Absenderadresse
- funktionierenden SMTP- oder anderen Mail-Transport
- SPF, DKIM und DMARC passend zur Absenderdomain

Ohne funktionierende Mailzustellung sind Passwort-Resets, Benachrichtigungen und manche Arbeitsabläufe unzuverlässig.

2.8. Hintergrundaufgaben

Für einen vollständigen Betrieb sollten Sie mindestens einplanen:

- Cron für den Laravel-Scheduler
- Queue-Worker für verzögerte Aufgaben
- Neustart der Worker nach Deployments

Wenn Sie Octane oder RoadRunner verwenden, kommen eigene Betriebsaufgaben wie Reloads und Health-Checks dazu.

3. Klassische Installation

Die klassische Installation ist der direkteste und transparenteste Weg. Sie eignet sich besonders, wenn Sie eine feste Linux-Umgebung mit Webserver, PHP und Datenbank selbst betreiben.

3.1. Zielbild

Sie erhalten:

- einen lokalen Pfarrplaner-Quellcode-Checkout
- installierte PHP- und npm-Abhängigkeiten
- gebaute Frontend-Assets
- eine fertige `.env`
- eine initialisierte Datenbank
- ein erstes Super-Admin-Konto

3.2. Server vorbereiten

Vor `git clone` sollten diese Punkte fertig sein:

- DNS-Eintrag und öffentliche URL
- HTTPS-Zertifikat
- Datenbank und Datenbankbenutzer
- Webserver-VHost auf `.../pfarrplaner/public`
- Systembenutzer, unter dem Webserver und Deployments arbeiten
- Schreibrechte für `storage/` und `bootstrap/cache/`

3.3. Quellcode holen

```
git clone https://codeberg.org/pfarr.tools/pfarrplaner.git
cd pfarrplaner
```

3.4. PHP-Abhängigkeiten installieren

```
composer install --no-dev --optimize-autoloader
```

Wenn Composer schon hier scheitert, sind fast immer PHP-Version oder PHP-Erweiterungen unvollständig.

3.5. Frontend-Abhängigkeiten und Assets

```
npm install
npm run build
```

Planen Sie dafür genügend Speicher und CPU-Zeit ein. Auf sehr kleinen Servern ist es oft sinnvoll, Assets in einer Build-Umgebung zu erzeugen und erst dann auf das Zielsystem zu übertragen.

3.6. Laufzeitverzeichnisse und Rechte

Typischerweise brauchen mindestens diese Verzeichnisse Schreibrechte:

- `storage/`
- `bootstrap/cache/`

Rechte sollten zum Webserver-Benutzer und zu Ihrem Deployment-Benutzer passen. Vermeiden Sie pauschale globale Schreibrechte, wenn gezieltere Rechte möglich sind.

3.7. Neuer CLI-Installer

Die produktive Erstinstallation erfolgt mit:

```
php artisan pfarrplaner:install
```

Der Installer fragt unter anderem ab:

- Anwendungsname
- öffentliche URL
- Anwendungsumgebung
- Debug-Modus
- Datenbanktyp und Datenbankzugang
- Mail-Absender
- erstes Super-Admin-Konto

Er schreibt die `.env`, erzeugt Schlüssel, führt Migrationen aus, seedet Rollen und legt das erste Super-Admin-Konto an.

3.8. Nicht-interaktive Installation

Für automatisierte Deployments kann der Installer auch mit Optionen aufgerufen werden, zum Beispiel:

```
php artisan pfarrplaner:install \
  --app-name="Pfarrplaner Musterstadt" \
  --app-url="https://planer.example.org" \
  --app-env=production \
  --app-debug=false \
  --db-connection=mysql \
  --db-host=127.0.0.1 \
  --db-port=3306 \
  --db-database=pfarrplaner \
  --db-username=pfarrplaner \
  --db-password="geheim" \
  --admin-first-name="Maria" \
  --admin-last-name="Muster" \
  --admin-email="admin@example.org" \
  --admin-password="sehr-geheim" \
  --mail-from-address="pfarrplaner@example.org" \
  --mail-from-name="Pfarrplaner"
```

3.9. Nacharbeiten nach dem Installer

Nach der Erstinstallation sollten Sie mindestens noch:

1. `php artisan optimize` ausführen.
2. Mailversand testen.
3. Login mit dem Super-Admin prüfen.
4. Scheduler und Queue-Worker einrichten.
5. Backup-Strategie festlegen.
6. Chromium-gestützte Ausgaben oder Screenshots testweise ausführen.

3.10. Scheduler und Queue

Für den Scheduler gehört ein Cronjob auf den Server, typischerweise einmal pro Minute:

```
* * * * * cd /pfad/zu/pfarrplaner && php artisan schedule:run >> /dev/null 2>&1
```

Falls Queue-Jobs genutzt werden, richten Sie zusätzlich einen dauerhaft laufenden Worker ein, zum Beispiel über `systemd` oder `supervisord`.

3.11. Webserver-Prüfliste

- `APP_URL` stimmt mit der echten öffentlichen URL überein.
- Der VHost zeigt auf `public/`.
- `APP_DEBUG` ist in produktiven Umgebungen ausdrücklich auf `false` gesetzt.
- Entwickler-Helfer wie Ignition-, Boost-, Telescope- oder Dusk-Endpunkte werden nur bei bewusst gesetzten Umgebungsvariablen aktiviert.
- PHP-FPM oder mod_php nutzt die richtige PHP-Version.
- Upload-Limits passen zu Dateianhängen und Importen.
- Timeouts sind nicht zu knapp.

3.12. Wann dieser Weg sinnvoll ist

Die klassische Installation ist meist die beste Wahl, wenn:

- Sie Linux, PHP und Webserver selbst verwalten.
- Sie Dateipfade und Systemdienste direkt kontrollieren möchten.
- Sie Updates bewusst und manuell einspielen wollen.

4. Installation mit Docker Compose

Dieser Weg ist für Umgebungen gedacht, in denen Pfarrplaner zusammen mit seinen Laufzeiten in Containern betrieben werden soll. Die inhaltliche Tiefe dieses Kapitels kann später noch ausgebaut werden; die Grundstruktur und der empfohlene Ablauf stehen bereits fest.

4.1. Zielbild

Eine Docker-Compose-Installation sollte mindestens diese Bestandteile sauber trennen:

- App-Container für PHP und die Laravel-Anwendung
- Datenbank-Container
- optional separater Webserver oder Reverse Proxy
- optional separater Queue-Worker
- optional separater Scheduler-Container oder Cron im App-Umfeld

4.2. Was in Docker trotzdem gleich bleibt

Auch im Containerbetrieb brauchen Sie weiterhin:

- eine gültige öffentliche URL
- HTTPS vor dem System
- persistente Datenbankdaten
- persistente Anwendungsdaten in `storage/`
- Mail-Konfiguration
- Chromium-Laufzeit für Puppeteer und Browsershot

4.3. Empfohlene Compose-Bausteine

- `app` : Pfarrplaner-Anwendung
- `db` : MySQL oder MariaDB
- `queue` : optional eigener Worker-Prozess
- `scheduler` : optional eigener Scheduler-Prozess
- `proxy` : optional Nginx, Traefik oder ein externer Reverse Proxy

4.4. Persistente Volumes

Mindestens diese Daten dürfen bei Container-Neustarts nicht verloren gehen:

- Datenbankdaten
- `storage/`
- gegebenenfalls Uploads, Caches oder exportierte Dateien, sofern sie betriebsrelevant sind

4.5. Build oder Image

Es gibt zwei übliche Wege:

1. Eigenes Image bauen, das Quellcode, Composer-Abhängigkeiten, Node-Build und Laufzeit bereits enthält.
2. Quellcode und Builds beim Start oder beim Deployment in den Container bringen.

Für produktive Umgebungen ist ein reproduzierbares Image meist die sauberere Wahl.

4.6. CLI-Installer im Container

Auch bei Docker Compose wird die Anwendung mit dem neuen CLI-Installer fertig eingerichtet, zum Beispiel:

```
docker compose exec app php artisan pfarrplaner:install
```

Oder nicht-interaktiv mit denselben Optionen wie bei der klassischen Installation.

4.7. Besondere Punkte bei Docker

- Dateirechte zwischen Host und Container müssen passen.
- Die öffentliche URL in `APP_URL` muss auf die echte Außenadresse zeigen, nicht auf einen internen Containernamen.
- Headless-Chromium braucht im Image die passenden Pakete und Bibliotheken.
- Queue- und Scheduler-Prozesse dürfen nicht davon abhängen, dass jemand interaktiv im Container angemeldet ist.

4.8. Empfohlene Reihenfolge

1. Compose-Datei und Volumes anlegen.
2. Container für App und Datenbank starten.
3. Erreichbarkeit der Datenbank aus dem App-Container prüfen.
4. `php artisan pfarrplaner:install` im App-Container ausführen.
5. Assets, Mail, Login und Hintergrundprozesse testen.

4.9. Noch gezielt zu ergänzen

Dieses Kapitel ist bewusst so aufgebaut, dass später zusätzliche Details leicht ergänzt werden können, zum Beispiel:

- konkrete Beispiel- `docker-compose.yml`
- Trennung von Build- und Runtime-Image
- Reverse-Proxy-Beispiele
- Health-Checks
- Backup von Volumes
- Rolling-Update-Strategien

5. Updates

Pfarrplaner kann sich in einem Release an mehreren Stellen ändern: PHP-Abhängigkeiten, npm-Abhängigkeiten, Assets, Migrationen, Views, Queue-Verhalten oder Laufzeitprozesse. Ein gutes Update besteht deshalb nie nur aus einem `git pull`.

5.1. Grundregel

Vor jedem Update brauchen Sie:

- aktuelles Backup der Datenbank
- Sicherung wichtiger Anwendungsdaten aus `storage/`
- kurze Prüfung offener lokaler Änderungen
- Klarheit, auf welchem Branch die produktive Instanz läuft

5.2. Updatepfad 1: klassische Installation

Für klassische Installationen ist der vorgesehene Weg der Update-Befehl:

```
php artisan install:updates
```

Im Hintergrund wird das npm-Skript `install:updates` ausgeführt. Dieses Skript prüft unter anderem:

- ob ein Upstream-Branch vorhanden ist
- ob lokale Änderungen existieren
- ob `composer install` nötig ist
- ob `npm install` nötig ist
- ob Assets neu gebaut werden müssen
- ob Migrationen auszuführen sind
- ob Caches, Queue und Octane neu gestartet werden müssen

5.2.1. Trockentest

Vor produktiven Updates ist diese Prüfung sehr sinnvoll:

```
php artisan install:updates --dry-run
```

Damit sehen Sie, welche Schritte ein Update auslösen würde, ohne sie schon auszuführen.

5.2.2. Typischer produktiver Ablauf

1. Backup erstellen.
2. `php artisan install:updates --dry-run`
3. Wartungsfenster oder kurzen Hinweis vorbereiten.
4. `php artisan install:updates`
5. Login, zentrale Ansichten, Mail und wichtige Berichte prüfen.

5.3. Updatepfad 2: Docker Compose

Im Containerbetrieb hängt das genaue Update von Ihrer Build-Strategie ab. Der sichere Grundablauf bleibt aber gleich:

1. Backup von Datenbank und persistenten Volumes erstellen.
2. Neues Image bauen oder ziehen.
3. Container mit der neuen Version bereitstellen.
4. Migrationen im App-Container ausführen.
5. Queue-Worker und Scheduler auf die neue Version umstellen.
6. Funktionstest durchführen.

Typische Befehle sehen zum Beispiel so aus:

```
docker compose pull
docker compose up -d
docker compose exec app php artisan migrate --force
docker compose exec app php artisan optimize
```

Wenn Sie Assets bereits im Image bauen, entfällt der Asset-Build auf dem Zielsystem. Wenn nicht, muss auch im Containerkontext ein neuer Frontend-Build Teil des Updates sein.

5.4. Rollback-Denken

Ein gutes Update ist erst dann sauber geplant, wenn Sie auch den Rückweg kennen.

Für klassische Installationen bedeutet das meist:

- vorherigen Git-Stand kennen
- Datenbank-Backup haben
- wissen, ob Migrationen rückwärts überhaupt sicher sind

Für Docker bedeutet das meist:

- vorheriges Image-Tag behalten
- alte Compose-Version oder Image-Referenz dokumentieren
- Datenbank-Backup vorhalten

5.5. Nach jedem Update prüfen

- Anmeldung
- Startseite
- Kalender
- Öffnen und Speichern eines Gottesdiensts
- ein Bericht oder Export
- Passwort-Reset oder Testmail
- Queue, Scheduler und Logs
- Chromium-basierte Ausgaben

6. Betrieb und Wartung

Nach der Installation beginnt die eigentliche Arbeit. Eine stabile Pfarrplaner-Instanz lebt davon, dass wiederkehrende Betriebsaufgaben zuverlässig erledigt werden.

6.1. Regelmäßige Aufgaben

- Logs prüfen
- Backups kontrollieren
- freien Speicherplatz überwachen
- Queue-Worker und Scheduler prüfen
- Mailzustellung beobachten
- Sicherheitsupdates des Betriebssystems einspielen

6.2. Wichtige Anwendungsbereiche im Betrieb

6.2.1. Konfiguration

Die zentrale Laufzeitkonfiguration liegt in der `.env`. Änderungen dort sollten bewusst dokumentiert werden. Nach relevanten Änderungen sind meist Cache- und Konfigurationsbefehle nötig, zum Beispiel `php artisan optimize` oder gezielte Cache-Clears.

6.2.2. Caches

Laravel und Vite arbeiten mit verschiedenen Caches. Nach Änderungen an Konfiguration, Views oder Assets ist es sinnvoll, den Cachezustand bewusst zu prüfen statt blind viele Befehle nacheinander auszuführen.

6.2.3. Queue

Sobald Pfarrplaner Hintergrundaufgaben nutzt, brauchen Sie eine überwachte Worker-Instanz. Ein abgestürzter Queue-Worker fällt im Alltag oft erst spät auf, wenn E-Mails, Exporte oder Folgeaktionen nicht mehr ankommen.

6.2.4. Scheduler

Der Scheduler sollte einmal pro Minute laufen. Fehlt dieser Cronjob oder ist er defekt, bleiben geplante Aufgaben liegen.

6.2.5. Dateispeicher

Anhänge, Bilder und andere Dateien dürfen nicht nur in Backups berücksichtigt werden. Sie brauchen auch ausreichend Speicherplatz und funktionierende Schreibrechte.

6.3. Prüfliste nach Server-Arbeiten

Wenn Sie am Server, an PHP oder an Docker-Grundlagen gearbeitet haben, prüfen Sie danach mindestens:

- Login
- Datei-Uploads
- Berichte und PDF-Ausgaben
- Mails
- Kalender- und Editoransichten

6.4. Rechte und Rollen in der Anwendung

Zur technischen Administration gehört auch ein klares Verständnis der administrativen Oberfläche:

- Benutzer
- Rollen
- Gemeinden
- Orte
- Vertretungs-Pools
- Lieder und liturgische Texte

Diese Inhalte werden funktional im Benutzerhandbuch beschrieben. Dieses Administratorhandbuch ergänzt dazu die Betriebs- und Sicherheitsaspekte.

7. Datenschutz

Dieses Kapitel beschreibt die Verarbeitung personenbezogener Daten in Pfarrplaner so vollständig wie möglich aus Sicht der Administration. Es soll helfen, das Verzeichnis der Verarbeitungstätigkeiten, Datenschutzinformationen, Berechtigungskonzepte und Löschregeln auf einer realen Installation sauber zu führen.

Pfarrplaner ist kein einzelner Datenbestand, sondern ein Arbeitswerkzeug für viele kirchliche Abläufe. Deshalb verarbeitet die Anwendung personenbezogene Daten an mehreren Stellen gleichzeitig: in Benutzerkonten, Gottesdiensten, Kasualien, Urlaubsabläufen, öffentlichen Freigaben, Berichten, Dateien und Systemprotokollen.

7.1. Wie dieses Kapitel zu lesen ist

Für jeden Verarbeitungsvorgang sind vor allem diese Fragen wichtig:

- Welche Daten werden verarbeitet?
- Zu welchem Zweck geschieht das?
- Wer darf die Daten sehen oder ändern?
- Werden Daten nach außen weitergegeben oder öffentlich angezeigt?
- Welche organisatorischen Regeln sollten Sie dafür festlegen?

Die rechtliche Bewertung selbst hängt von Ihrer kirchlichen oder staatlichen Datenschutzordnung ab. Dieses Kapitel ersetzt keine juristische Prüfung, beschreibt aber die tatsächlichen Datenflüsse in der Anwendung.

7.2. Grundsatz: Wo in Pfarrplaner personenbezogene Daten vorkommen

Pfarrplaner verarbeitet personenbezogene Daten in diesen großen Bereichen:

- Benutzerkonten und Personenstammdaten
- Rollen, Rechte, Gemeindezuordnungen und Startseiten-Einstellungen
- Gottesdienste, Veranstaltungen und Mitwirkende
- Predigt- und Liturgievorbereitung, soweit Personenbezug enthalten ist
- Taufen, Trauungen und Beerdigungen
- Abwesenheiten, Vertretungen und Pool-Zuständigkeiten
- Anmeldungen, Platzreservierungen und Kontaktlisten
- Kommentare, Dateien und automatisch erzeugte Berichte
- öffentliche Seiten, Freigabelinks und Streaming-Angaben
- technische Protokolle, Backups, Mailversand und API-Zugänge

7.3. 1. Benutzerkonten und Personenstammdaten

7.3.1. Welche Daten verarbeitet werden

In Benutzer- und Personenkonten kommen je nach Nutzung unter anderem vor:

- Titel
- Vorname und Nachname
- E-Mail-Adresse
- Passwort beziehungsweise Passwort-Reset-Status
- Telefonnummer
- Adresse
- Benutzerbild
- Rollen
- Gemeindezuordnungen
- Pfarramtszuordnungen
- Menü- und Startseiten-Einstellungen
- Benachrichtigungseinstellungen
- Urlaubs- und Vertretungsrelevanz
- API- oder Zugangstoken, soweit genutzt

7.3.2. Zweck

- Anmeldung an der Anwendung
- Rechteprüfung
- Anzeige von Zuständigkeiten und Kontaktmöglichkeiten
- interne Kommunikation und Benachrichtigung
- Planung von Diensten, Urlaub und Vertretungen
- Personalisierung der Oberfläche

7.3.3. Wer Zugriff hat

- Administratorinnen und Administratoren
- Personen mit passenden Rechten in der Benutzerverwaltung
- in Teilbereichen andere berechtigte Nutzerinnen und Nutzer, wenn Namen, Rollen oder Kontaktdaten in Planungsansichten erscheinen

7.3.4. Datenschutz-Hinweise für die Administration

- Vergeben Sie Administrationsrechte sparsam.
- Prüfen Sie, ob wirklich alle Benutzerkonten Adress- und Telefondaten brauchen.
- Halten Sie gesperrte und gelöschte Konten organisatorisch auseinander.
- Dokumentieren Sie, wie mit ehemaligen Mitarbeitenden verfahren wird.

7.4. 2. Rollen, Rechte und Gemeindezuordnungen

7.4.1. Welche Daten verarbeitet werden

Pfarrplaner speichert nicht nur Identitätsdaten, sondern auch Zuordnungen, aus denen sich ein personenbezogenes Profil ergibt:

- Rolle einer Person
- Schreib- oder Leserechte in Gemeinden
- Benachrichtigungsregeln
- Zuständigkeit für Urlaub, Prüfung oder Genehmigung
- Sichtbarkeit von Menüpunkten und Startseiten-Reitern

7.4.2. Zweck

- Steuerung des Datenzugriffs
- Abbildung organisatorischer Zuständigkeiten
- Anzeige arbeitsrelevanter Inhalte

7.4.3. Datenschutz-Hinweise für die Administration

- Rollen und Detailrechte sollten regelmäßig geprüft werden.
- Sichtbarkeit im Menü ersetzt keine Rechteprüfung und darf organisatorisch nicht missverstanden werden.
- Rechte sollten immer an tatsächliche Aufgaben gebunden sein, nicht an Bequemlichkeit.

7.5. 3. Gottesdienste, Veranstaltungen und Mitwirkende

7.5.1. Welche Daten verarbeitet werden

In Gottesdiensten und Veranstaltungen können personenbezogene Daten vieler Beteiligter stehen, zum Beispiel:

- Namen der Pfarrpersonen, Organist:innen, Mesner:innen und weiterer Mitwirkender
- dienstbezogene Zuordnungen
- Kommentare und interne Hinweise
- predigt- oder liturgierelevante Notizen mit Personenbezug
- Telefonnummer für telefonische Anmeldung
- Ansprechpartnerdaten in Einzelfällen
- Dateien und Anhänge

7.5.2. Zweck

- Planung und Durchführung von Gottesdiensten und Veranstaltungen
- interne Abstimmung
- Vorbereitung von Ausgaben, Plänen und Berichten

7.5.3. Wer Zugriff hat

- Personen mit Zugriffsrechten auf die betroffenen Gemeinden und Gottesdienste
- in Sonderfällen Empfänger öffentlicher oder halböffentlicher Ausgaben

7.5.4. Datenschutz-Hinweise für die Administration

- Prüfen Sie regelmäßig, welche Felder wirklich für öffentliche Ausgaben vorgesehen sind.
- Interne Kommentare dürfen nicht mit öffentlichen Beschreibungen verwechselt werden.
- Datei-Anhänge können wesentlich sensibler sein als die sichtbaren Stammdaten eines Gottesdiensts.

7.6. 4. Predigten, Liturgie und vorbereitende Inhalte

7.6.1. Welche Daten verarbeitet werden

Dieser Bereich wirkt auf den ersten Blick nicht stark personenbezogen, kann aber personenbezogene Inhalte enthalten:

- Namen verantwortlicher Personen in Liturgieelementen
- Predigtnotizen mit Bezug auf konkrete Personen
- interne Vorbereitungstexte
- importierte oder angehängte Dateien mit Personenbezug

7.6.2. Zweck

- Vorbereitung und Durchführung von Gottesdiensten
- Zusammenarbeit zwischen Beteiligten

7.6.3. Datenschutz-Hinweise für die Administration

- Schulen Sie Nutzerinnen und Nutzer darin, sensible Seelsorge- oder Gesprächsinhalte nicht unbedacht in freie Textfelder zu schreiben.
- Prüfen Sie Berechtigungen für Liturgie- und Predigtbereiche besonders sorgfältig.

7.7. 5. Taufen

7.7.1. Welche Daten verarbeitet werden

Bei Taufen werden je nach Fall verarbeitet:

- Name des Täuflings
- Geburtsdatum
- Geburtsort
- Adresse
- Postleitzahl und Wohnort
- Telefonnummer
- E-Mail-Adresse
- Pronomen
- Kirchengemeinde
- Erstkontakt mit Datum und Kontaktperson
- Angaben zur Dimissoriale
- Taufgespräch, Taufspruch und Vorbereitung
- Urkunden- und Kirchenbuchstatus
- Dateien und Unterlagen

7.7.2. Zweck

- Vorbereitung, Durchführung und Nacharbeit einer Taufe
- Verbindung mit einem Taufgottesdienst
- Dokumentation des kirchlichen Verwaltungsvorgangs

7.7.3. Besondere Sensibilität

Taufdaten betreffen oft Kinder und Familien. Damit ist besondere Zurückhaltung wichtig bei:

- Zugriffsrechten
- Dateianhängen
- Exporten und Ausdrucken
- Speicherfristen in Arbeitsdateien

7.8. 6. Trauungen

7.8.1. Welche Daten verarbeitet werden

Bei Trauungen verarbeitet Pfarrplaner insbesondere:

- Namen beider Ehepartner:innen
- gegebenenfalls Geburtsnamen
- E-Mail-Adressen
- Telefonnummern
- Angaben zum Traugespräch
- Trautext
- Dimissoriale-Status je Person
- Genehmigungsstatus
- Urkunden- und Kirchenbuchstatus
- Dateien und Unterlagen

7.8.2. Zweck

- Vorbereitung, Durchführung und Nacharbeit einer Trauung
- organisatorische und kirchenrechtliche Begleitung

7.8.3. Datenschutz-Hinweise für die Administration

- Prüfen Sie besonders, wer Zugriff auf Dateien und Begleitunterlagen hat.
- Klären Sie intern, welche Unterlagen dauerhaft im System bleiben sollen und welche nur vorübergehend hochgeladen werden.

7.9. 7. Beerdigungen

7.9.1. Welche Daten verarbeitet werden

Bei Beerdigungen ist der Personenbezug besonders weitreichend. Typische Daten sind:

- Name der verstorbenen Person
- Geburtsname
- Rufname
- Geburtsdatum und Sterbedatum
- Geburtsort und Sterbeort
- Adresse
- Bestattungsart
- Kirchenzugehörigkeit und kirchliche Zusatzangaben
- Daten zur wichtigsten Kontaktperson oder zu Angehörigen
- Kontaktdaten von Angehörigen
- Trauergespräch
- Abkündigung
- Predigttext und vorbereitende Notizen
- Dimissoriale-Angaben
- Kirchenbuchstatus
- Dateien, Formulare und weitere Unterlagen

7.9.2. Zweck

- Vorbereitung und Durchführung einer Bestattung
- Kommunikation mit Angehörigen und beteiligten Stellen
- Dokumentation kirchlicher Verwaltungsabläufe

7.9.3. Besondere Sensibilität

Beerdigungsdaten gehören zu den sensibelsten Daten im System. Administratorisch wichtig sind deshalb:

- strenge Rechtevergabe
- sehr bewusster Umgang mit Exporten
- regelmäßige Prüfung alter Dateien
- klare Regeln für lokale Kopien und Ausdrucke

7.10. 8. Abwesenheiten, Urlaub und Vertretungen

7.10.1. Welche Daten verarbeitet werden

Im Urlaubs- und Vertretungsbereich werden verarbeitet:

- Name der betroffenen Person
- Abwesenheitszeiträume
- Workflow-Status
- zuständige prüfende oder genehmigende Personen
- Vertretungspersonen
- Vertretungshinweise
- Pool- und Poolmaster-Zeiten
- gegebenenfalls Dateien

7.10.2. Zweck

- Personal- und Vertretungsplanung
- Genehmigungs- und Prüfabläufe
- Erreichbarkeit während Abwesenheiten

7.10.3. Öffentliche oder externe Sichtbarkeit

Ein Teil dieser Daten kann über Pool- oder Vertretungsübersichten sichtbar werden. Je nach Konfiguration erscheinen dort zum Beispiel:

- Name
- Amt
- Telefonnummer
- E-Mail-Adresse
- Vertretungszeitraum

7.10.4. Datenschutz-Hinweise für die Administration

- Prüfen Sie genau, welche Abwesenheitsdaten intern bleiben und welche auf öffentlichen Übersichten erscheinen dürfen.
- Dokumentieren Sie, ob Vertretungsseiten nur intern oder auch für Dritte wie Bestatter gedacht sind.

7.11. 9. Teams, Pools und Kontaktstellen

7.11.1. Welche Daten verarbeitet werden

In Teams und Pools verarbeitet Pfarrplaner:

- Namen von Mitgliedern
- Gemeindezuordnungen
- Ansprechpartner:innen
- Telefonnummern
- E-Mail-Adressen
- Institution oder Amt

7.11.2. Zweck

- Zuordnung von Gruppen
- Vertretungsorganisation
- Kontaktaufnahme über zentrale Stellen oder konkrete Personen

7.11.3. Datenschutz-Hinweise für die Administration

- Prüfen Sie, ob besser zentrale Kontaktstellen statt privater Personendaten angezeigt werden sollen.
- Halten Sie öffentliche und interne Pools organisatorisch getrennt.

7.12. 10. Anmeldungen, Platzreservierungen und Kontaktlisten

7.12.1. Welche Daten verarbeitet werden

Bei Anmeldungen und Sitzplatzverwaltung können personenbezogene Daten verarbeitet werden wie:

- Name der anmeldenden oder teilnehmenden Person
- Telefonnummer für Rückfragen oder telefonische Anmeldung
- Reservierungs- oder Sitzplatzdaten
- Teilnehmerzahlen mit möglichem Personenbezug

7.12.2. Zweck

- Organisation von Veranstaltungen
- Teilnahmebegrenzungen
- Nachverfolgung von Reservierungen

7.12.3. Datenschutz-Hinweise für die Administration

- Klären Sie intern, wie lange Anmeldedaten nach einer Veranstaltung noch gebraucht werden.
- Prüfen Sie besonders Exportlisten und Ausdrucke, weil sie leicht außerhalb des Systems weitergegeben werden.

7.13. 11. Kommentare, Dateien und Anhänge

7.13.1. Welche Daten verarbeitet werden

In vielen Bereichen können Dateien oder Kommentare gespeichert werden:

- Kommentare zu Gottesdiensten und Vorgängen
- hochgeladene Dateien zu Kasualien, Urlaub, Veranstaltungen und weiteren Objekten
- automatisch erzeugte Formulare
- eingescannte Unterlagen

7.13.2. Zweck

- interne Zusammenarbeit
- Dokumentation
- Vorbereitung und Ablage von Unterlagen

7.13.3. Besondere Risiken

Kommentare und Dateien sind datenschutzrechtlich oft kritischer als die sichtbaren Listenfelder, weil dort sehr leicht:

- freie personenbezogene Notizen
- Gesundheits- oder Familiendaten
- unterschriebene Formulare
- Schriftwechsel

gespeichert werden können.

7.13.4. Datenschutz-Hinweise für die Administration

- Definieren Sie klare Regeln, welche Unterlagen ins System gehören.
- Prüfen Sie Download-Rechte.
- Denken Sie bei Löschkonzepten immer auch an Anhänge mit.

7.14. 12. Berichte, Exporte und erzeugte Dokumente

7.14.1. Welche Daten verarbeitet werden

Pfarrplaner erzeugt PDF-, Word-, Excel-, CSV- und HTML-Ausgaben. Darin können nahezu alle im System vorhandenen personenbezogenen Daten wieder auftauchen, zum Beispiel:

- Namen und Dienstzuordnungen
- Kontaktlisten
- Kasualdaten
- Abwesenheits- oder Vertretungsangaben
- Anmeldungen

7.14.2. Zweck

- Arbeitslisten
- Auswertungen
- Druckausgaben
- Weitergabe an berechtigte Stellen

7.14.3. Datenschutz-Hinweise für die Administration

- Prüfen Sie nicht nur, wer im System sehen darf, sondern auch, wer Exporte erzeugen darf.
- Legen Sie fest, wie lange exportierte Dateien außerhalb des Systems aufbewahrt werden.
- Schulen Sie Nutzerinnen und Nutzer im sicheren Umgang mit lokal gespeicherten Dateien.

7.15. 13. Öffentliche Seiten, Freigabelinks und Streaming

7.15.1. Welche Daten verarbeitet oder veröffentlicht werden

Pfarrplaner kann personenbezogene Daten auf öffentlichen oder halböffentlichen Seiten anzeigen, etwa:

- Namen zuständiger Pfarrpersonen
- Telefonnummern und E-Mail-Adressen
- Vertretungs- und Poolinformationen
- Kontaktlisten zu Gottesdiensten
- Daten zu gestreamten Gottesdiensten
- YouTube- oder Kinderkirche-Streaming-Links

7.15.2. Zweck

- Information der Öffentlichkeit
- Erreichbarkeit
- Einbindung externer Darstellungen

7.15.3. Datenschutz-Hinweise für die Administration

- Öffentliche Seiten müssen besonders streng geprüft werden.
- Nutzen Sie möglichst dienstliche statt private Kontaktdaten.
- Kontrollieren Sie Freigabelinks regelmäßig.
- Prüfen Sie, ob öffentliche Darstellungen wirklich nur die minimal nötigen Daten zeigen.

7.16. 14. Kalenderverbindungen, Benachrichtigungen und externe Dienste

7.16.1. Welche Daten verarbeitet werden

Je nach Einrichtung können Daten aus Pfarrplaner nach außen fließen oder von außen eingebunden werden, zum Beispiel über:

- Kalenderverbindungen
- Benachrichtigungs-E-Mails
- YouTube-Streaming
- KonfiApp oder CommuniApp
- eingebettete oder freigegebene Inhalte

7.16.2. Zweck

- technische Integration
- Veröffentlichung
- Kommunikation

7.16.3. Datenschutz-Hinweise für die Administration

- Dokumentieren Sie jede aktive Integration separat.
- Prüfen Sie, welche Daten an welchen externen Dienst übergeben werden.
- Verwenden Sie nur nötige Zugangsdaten und speichern Sie sie geschützt.

7.17. 15. API-Zugänge, Tokens und technische Identifikatoren

7.17.1. Welche Daten verarbeitet werden

Im technischen Betrieb kommen zusätzlich personenbezogene oder personenbeziehbare Daten vor:

- E-Mail-Adresse als Benutzername
- API-Tokens
- Passwort-Reset-Vorgänge
- Benutzerbezug in Änderungs- oder Workflow-Aktionen
- gegebenenfalls IP-Adressen in technischen Logs

7.17.2. Zweck

- Authentifizierung
- technische Absicherung
- Nachvollziehbarkeit von Zugriffen

7.17.3. Datenschutz-Hinweise für die Administration

- API-Tokens nur gezielt vergeben und regelmäßig prüfen.
- Passwörter nie außerhalb sicherer Prozesse weitergeben.
- Logfiles mit Personenbezug nicht länger als nötig aufbewahren.

7.18. 16. Protokolle, Backups und Wiederherstellung

7.18.1. Welche Daten verarbeitet werden

Auch wenn Nutzerinnen und Nutzer sie nicht direkt sehen, enthalten diese Bereiche regelmäßig personenbezogene Daten:

- Anwendungslogs
- Mail- und Fehlerlogs
- Datenbank-Backups
- Dateisicherungen aus `storage/`
- temporäre Exporte oder Renderdateien

7.18.2. Zweck

- Fehleranalyse
- Betriebssicherheit
- Wiederherstellung nach Störung

7.18.3. Datenschutz-Hinweise für die Administration

- Backups sind keine datenschutzfreien Kopien, sondern vollständige oder teilweise Duplikate des Systems.
- Sichern Sie Backups genauso streng wie das Livesystem.
- Definieren Sie Aufbewahrungsfristen und Zugriffsrechte.

7.19. 17. Organisatorische Prüfliste für Administrator:innen

Für eine produktive Pfarrplaner-Instanz sollten Sie mindestens diese Punkte dokumentiert haben:

1. Welche Benutzergruppen welche Daten sehen dürfen.
2. Welche öffentlichen Seiten aktiv sind.
3. Welche externen Dienste angebunden sind.
4. Welche Dateien und Exporte mit personenbezogenen Daten typischerweise entstehen.
5. Wer Backups lesen oder wiederherstellen darf.
6. Wie lange alte Konten, Anhänge, Logs und Exporte aufbewahrt werden.
7. Wie Berichtigungen, Sperrungen und Löschungen organisatorisch abgewickelt werden.

7.20. Was dieses Kapitel bewusst nicht festlegt

Dieses Kapitel beschreibt die tatsächlichen Verarbeitungsvorgänge in Pfarrplaner, legt aber nicht selbst fest:

- welche Rechtsgrundlage jeweils gilt
- welche Aufbewahrungsfrist im Einzelfall richtig ist
- welche kirchliche oder staatliche Datenschutzordnung für Ihre Einrichtung maßgeblich ist

Diese Entscheidungen müssen Sie für Ihre Organisation zusätzlich sauber dokumentieren.

8. Sicherheit, Backup und Datenschutz

Pfarrplaner verarbeitet personenbezogene Daten. Darum gehören Sicherheit und Wiederherstellbarkeit nicht an den Rand, sondern in den normalen Betriebsalltag.

8.1. Zugangsschutz

- Nur HTTPS nach außen
- starke Passwörter
- möglichst wenige Super-Admin-Konten
- getrennte Konten für normale Nutzung und Administration, wenn sinnvoll
- keine gemeinsamen Dauerpasswörter im Team

8.2. Serverhärtung

- nur nötige Ports öffnen
- SSH absichern
- Sicherheitsupdates des Betriebssystems zeitnah einspielen
- keine unnötigen Werkzeuge auf dem Produktionsserver belassen

8.3. Pfarrplaner-spezifische Härtung

Für Pfarrplaner selbst sollten Sie besonders darauf achten:

- `APP_DEBUG` in Produktion auf `false`
- Entwicklerhilfen wie Boost, Telescope, Ignition-Runnable-Solutions oder Dusk nur gezielt und nie dauerhaft in Produktion aktivieren
- öffentliche Datei- und Bildlinks aus `attachments/` nur über signierte URLs weitergeben
- keine alten Direktlinks zu Logout-, Token-, Patch- oder Benutzerwechsel-Routen weiterverwenden, wenn Integrationen oder Eigenanpassungen existieren
- öffentliche Einbettungen und Feeds regelmäßig darauf prüfen, ob sie wirklich nur die beabsichtigten Daten preisgeben

Ein technischer Überblick über den Sicherheitsdurchgang und die abgesicherten Routen liegt im Technikhandbuch.

8.4. Backup

Ein brauchbares Backup umfasst mindestens:

- Datenbank
- `storage/`
- falls nötig `.env` oder andere sicher verwahrte Konfigurationsdaten

Wichtig ist nicht nur das Erstellen, sondern auch das Wiederherstellen. Testen Sie regelmäßig, ob sich aus den Sicherungen wirklich eine lauffähige Instanz herstellen lässt.

8.5. Datenschutz

Prüfen Sie besonders:

- Wer darf welche personenbezogenen Daten sehen?
- Welche öffentlichen Ausgaben enthalten personenbezogene Daten?
- Welche Demo-, Test- oder Schulungsinstanzen enthalten Produktivdaten?
- Wie lange bleiben Protokolle, Uploads und Exporte gespeichert?

8.6. Mail und externe Integrationen

Externe Integrationen sollten sparsam und nachvollziehbar konfiguriert sein. Zugangsdaten gehören nur in geschützte Konfigurationswege, nicht in Tickets, Chats oder unverschlüsselte Notizen.

8.7. Vor größeren Änderungen

Vor Updates, Serverumzügen oder Integrationsarbeiten:

1. Backup prüfen.
2. Wartungsfenster festlegen.
3. Rückfallplan festhalten.
4. Verantwortliche informieren.

9. Fehlersuche

Bei Störungen lohnt es sich, zuerst den betroffenen Bereich einzugrenzen: Anwendung, Datenbank, Mail, Queue, Browser-Laufzeit oder Webserver.

9.1. Häufige Fehlerbilder

9.1.1. Login funktioniert nicht

Prüfen Sie:

- stimmt `APP_URL` ?
- stimmen Session- und Cookie-Einstellungen?
- funktioniert die Datenbank?
- ist das Benutzerkonto aktiv?

9.1.2. Seiten laden, aber Speichern schlägt fehl

Prüfen Sie:

- Anwendungslog
- Datenbankverbindung
- PHP-Fehler
- fehlende Schreibrechte

9.1.3. Debug- oder Entwicklerseiten sind erreichbar

Prüfen Sie:

- `APP_DEBUG` steht nicht versehentlich auf `true`
- `BOOST_ENABLED` und `BOOST_BROWSER_LOGS_WATCHER` sind nicht unbeabsichtigt aktiviert
- `IGNITION_ENABLE_RUNNABLE_SOLUTIONS` ist nicht aktiviert
- `TELESCOPE_ENABLED` ist nur für bewusst abgesicherte lokale Sitzungen aktiv
- `DUSK_ENABLED` ist nur für gezielte Browser-Tests aktiv
- Entwicklungswerkzeuge wie Dusk oder Telescope laufen nicht auf einem öffentlich erreichbaren System

9.1.4. Mails kommen nicht an

Prüfen Sie:

- Mail-Konfiguration in `.env`
- SMTP-Verbindung
- SPF, DKIM, DMARC
- Queue-Worker, falls Mailversand asynchron läuft

9.1.5. PDF- oder Browser-Ausgaben schlagen fehl

Prüfen Sie:

- Chromium/Chrome installiert?
- Systembibliotheken vollständig?
- Startet der Browser für den Webserver-Benutzer?
- gibt es genug temporären Speicher?

9.1.6. Updates brechen ab

Prüfen Sie:

- lokaler Git-Status
- Upstream-Branch
- Composer- oder npm-Abhängigkeiten
- Migrationen
- Dateirechte

9.2. Wichtige Protokollquellen

- Laravel-Logs in `storage/logs/`
- Webserver-Logs
- PHP-FPM- oder Prozess-Logs
- Queue- und Scheduler-Logs
- Container-Logs bei Docker

9.3. Gute Reihenfolge bei der Fehlersuche

1. Symptom kurz beschreiben.
2. Reproduzierbarkeit prüfen.
3. Logs derselben Zeitspanne ansehen.
4. zuletzt geänderte Systemteile notieren.
5. erst dann korrigieren oder zurückrollen.

10. Stichwortverzeichnis

Dieses Stichwortverzeichnis hilft beim schnellen Finden typischer Administrationsaufgaben.

10.1. A

10.2. B

10.3. C

10.4. D

10.5. E

10.6. F

10.7. G

10.8. H

- HTTPS: Systemvoraussetzungen, Sicherheit, Backup und Datenschutz

10.9. I

- Installation: Klassische Installation, Installation mit Docker Compose

10.10. K

- Konfiguration: Klassische Installation, Betrieb und Wartung

10.11. L

- Logs: Betrieb und Wartung, Fehlersuche

10.12. M

- Mailversand: Systemvoraussetzungen, Klassische Installation, Fehlersuche
- Migrationen: Klassische Installation, Updates

10.13. N

- Node.js: Systemvoraussetzungen, Klassische Installation
- npm: Systemvoraussetzungen, Klassische Installation, Updates

10.14. P

- Passwort-Reset: Systemvoraussetzungen, Updates
- PHP 8.4: Systemvoraussetzungen
- Puppeteer: Systemvoraussetzungen, Installation mit Docker Compose

10.15. Q

- Queue: Systemvoraussetzungen, Klassische Installation, Betrieb und Wartung, Fehlersuche

10.16. R

- Rollback: Updates
- Rollen und Rechte: Betrieb und Wartung

10.17. S

- Scheduler: Systemvoraussetzungen, Klassische Installation, Betrieb und Wartung
- Sicherheit: Sicherheit, Backup und Datenschutz
- SMTP: Systemvoraussetzungen, Fehlersuche
- Storage: Klassische Installation, Installation mit Docker Compose, Sicherheit, Backup und Datenschutz

10.18. U

10.19. V

10.20. W