

# Sauberfix - Terminverwaltung für Reinigungsunternehmen

Eine webbasierte Anwendung zur Verwaltung von Kunden, Mitarbeitern und Terminen für Reinigungsunternehmen.

## Inhaltsverzeichnis

1. Überblick
  2. Technologie-Stack
  3. Installation
  4. Konfiguration
  5. Funktionen
  6. Benutzerrollen
  7. API-Dokumentation
  8. Datenmodell
  9. Sicherheit
  10. Deployment
- 

## Überblick

**Sauberfix** ist ein umfassendes Terminverwaltungssystem für Reinigungsunternehmen. Die Anwendung ermöglicht:

- Verwaltung von Kunden mit Kontaktdaten
  - Verwaltung von Mitarbeitern mit Rollen
  - Planung und Übersicht von Reinigungsterminen
  - Kalenderansicht mit Drag & Drop
  - Automatische Terminerinnerungen (24 Stunden vorher)
  - Kollisionserkennung bei Terminüberschneidungen
- 

## Technologie-Stack

Komponente	Technologie
Backend	ASP.NET Core 9.0 (Minimal APIs)
Sprache	Visual Basic .NET + C#
Datenbank	PostgreSQL
ORM	Entity Framework Core 9.0
Authentifizierung	JWT Bearer Tokens
Passwort-Hashing	BCrypt (WorkFactor 12)
Frontend	HTML5, JavaScript, CSS
Kalender	FullCalendar Scheduler 6.1.19
Container	Docker
Orchestrierung	Kubernetes (K3s)

---

## Installation

### Voraussetzungen

- .NET 9.0 SDK

- PostgreSQL Datenbank
- Node.js (optional, für Frontend-Entwicklung)

## Lokale Entwicklung

### 1. Repository klonen

```
git clone https://github.com/fabian-lindhardt/sauberfix.git
cd sauberfix
```

### 2. Datenbank erstellen

```
CREATE DATABASE sauberfix;
CREATE USER appuser WITH PASSWORD 'IhrPasswort';
GRANT ALL PRIVILEGES ON DATABASE sauberfix TO appuser;
```

### 3. Konfiguration anpassen

Erstellen Sie `appsettings.Development.json`:

```
{
  "ConnectionStrings": {
    "DefaultConnection": "Host=localhost;Port=5432;Database=sauberfix;Username=appuser;Password=IhrPasswort"
  },
  "JwtSettings": {
    "Key": "IhrSichererSchluesselMitMindestens32Zeichen!",
    "Issuer": "SauberfixAPI",
    "Audience": "SauberfixClient"
  }
}
```

### 4. Anwendung starten

```
dotnet run
```

### 5. Browser öffnen

- Hauptanwendung: `http://localhost:5000`
- Kalenderansicht: `http://localhost:5000/scheduler.html`

## Standard-Anmeldedaten

Bei der ersten Ausführung wird automatisch ein Admin-Benutzer erstellt:

Feld	Wert
Benutzername	admin
Passwort	admin123

**Wichtig:** Ändern Sie das Passwort nach der ersten Anmeldung!

---

## Konfiguration

### Umgebungsvariablen

Die Anwendung kann über Umgebungsvariablen konfiguriert werden:

Variable	Beschreibung
ConnectionStrings__DefaultConnection	PostgreSQL Verbindungsstring
JwtSettings__Key	Geheimer Schlüssel für JWT-Signierung (min. 32 Zeichen)
JwtSettings__Issuer	JWT Aussteller
JwtSettings__Audience	JWT Zielgruppe
ASPNETCORE_ENVIRONMENT	Umgebung (Development/Production)

## CORS-Konfiguration

Die erlaubten Origins sind in `Program.vb` definiert:

- `localhost` - Lokale Entwicklung
- `coder.flairtec.de` - Coder Workspace
- `sauberfix.flairtec.de` - Produktion

## Funktionen

### 1. Kundenverwaltung

- Kunden anlegen mit Name, Firma, Adresse, E-Mail und Telefon
- Ortsnormalisierung (PLZ/Stadt werden wiederverwendet)
- Kunden bearbeiten und löschen
- Pflichtfelder: Vorname, Nachname, PLZ, Stadt

### 2. Mitarbeiterverwaltung

- Mitarbeiter mit Benutzername und Passwort anlegen
- Rollen zuweisen (Admin oder User)
- Sichere Passwortspeicherung mit BCrypt
- Mitarbeiter bearbeiten und löschen

### 3. Terminverwaltung

- Termine mit Start- und Endzeit erstellen
- Kunde und Mitarbeiter zuweisen
- Status: Geplant, Erledigt, Storniert
- **Kollisionserkennung:** Verhindert Doppelbuchungen
- Termine bearbeiten und löschen

### 4. Kalenderansicht (Scheduler)

- Wochenübersicht mit allen Mitarbeitern als Ressourcen
- Drag & Drop zum Verschieben von Terminen
- Größe ändern zum Anpassen der Dauer
- Klick auf leeren Slot zum Erstellen
- Farbcodierung:
  - **Blau:** Erinnerung ausstehend
  - **Grün:** Erinnerung gesendet

### 5. Automatische Erinnerungen

- Hintergrunddienst prüft alle 60 Sekunden
- Erkennt Termine die in 24 Stunden stattfinden
- Markiert Termine als "Erinnerung gesendet"

- Nur für Termine mit Status “Geplant”

---

## Benutzerrollen

### Admin

Berechtigung	Zugriff
Alle Termine sehen	Ja
Termine erstellen/bearbeiten/löschen	Ja
Kunden verwalten	Ja
Mitarbeiter verwalten	Ja
Andere Admins erstellen	Ja

### User (Mitarbeiter)

Berechtigung	Zugriff
Eigene Termine sehen	Ja
Alle Termine sehen	Nein
Termine erstellen/bearbeiten/löschen	Nein
Kunden verwalten	Nein
Mitarbeiter verwalten	Nein

---

## API-Dokumentation

Alle Endpunkte (außer /login) erfordern einen gültigen JWT-Token im Authorization-Header:

Authorization: Bearer <token>

### Authentifizierung

**POST /login** Benutzer anmelden und Token erhalten.

**Request:**

```
{
  "username": "admin",
  "password": "admin123"
}
```

**Response:**

```
{
  "token": "eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9...",
  "username": "admin",
  "rolle": "Admin"
}
```

### Kunden

**GET /kunden** Alle Kunden abrufen.

**POST /kunden** Neuen Kunden erstellen.

**Request:**

```
{
  "vorname": "Max",
  "nachname": "Mustermann",
  "firma": "Muster GmbH",
  "strasse": "Musterstraße 1",
  "plz": "12345",
  "stadt": "Musterstadt",
  "email": "max@muster.de",
  "telefon": "0123-456789"
}
```

**PUT /kunden/{id}** Kunden aktualisieren.

**DELETE /kunden/{id}** Kunden löschen.

**Mitarbeiter**

**GET /mitarbeiter** Alle Mitarbeiter abrufen.

**POST /mitarbeiter** Neuen Mitarbeiter erstellen.

**Request:**

```
{
  "username": "mmuster",
  "password": "sicheresPasswort123",
  "vorname": "Maria",
  "nachname": "Muster",
  "rolle": "User"
}
```

**PUT /mitarbeiter/{id}** Mitarbeiter aktualisieren (Passwort optional).

**DELETE /mitarbeiter/{id}** Mitarbeiter löschen.

**Termine**

**GET /termine** Termine abrufen. - **Admin:** Alle Termine - **User:** Nur eigene Termine

**POST /termine** Neuen Termin erstellen.

**Request:**

```
{
  "datumUhrzeit": "2025-01-15T09:00:00",
  "endzeit": "2025-01-15T11:00:00",
  "beschreibung": "Büroreinigung",
  "kundeId": 1,
  "mitarbeiterId": 2
}
```

**Fehler bei Kollision:**

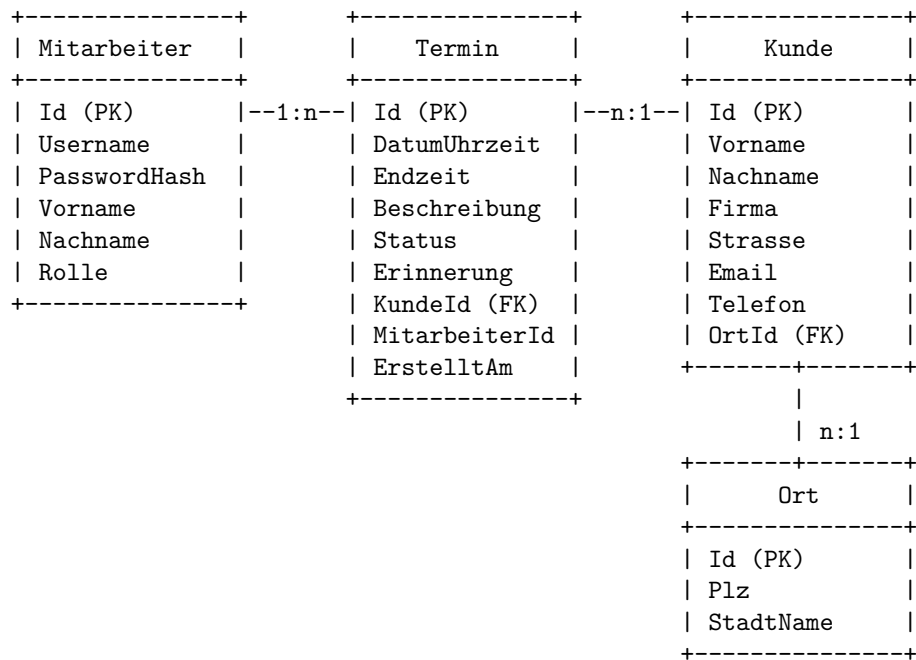
```
{
  "type": "...",
  "title": "An error occurred",
  "detail": "Der Mitarbeiter hat bereits einen Termin zu dieser Zeit"
}
```

**PUT** /termine/{id} Termin aktualisieren.

**DELETE** /termine/{id} Termin löschen.

## Datenmodell

### Entity-Relationship-Diagramm



### Termin-Status

Status	Bedeutung
Geplant	Termin ist geplant und aktiv
Erledigt	Termin wurde durchgeführt
Storniert	Termin wurde abgesagt

## Sicherheit

### Implementierte Maßnahmen

Maßnahme	Status	Beschreibung
BCrypt Passwort-Hashing	Ja	WorkFactor 12 (4096 Iterationen)

Maßnahme	Status	Beschreibung
JWT Token Authentifizierung	Ja	8 Stunden Gültigkeit
HTTPS Erzwingung	Ja	RequireHttpsMetadata=True
CORS Einschränkung	Ja	Nur erlaubte Domains
XSS-Schutz	Ja	HTML-Escaping im Frontend
Rollenbasierte Zugriffskontrolle	Ja	Admin/User Unterscheidung
SQL Injection Schutz	Ja	Entity Framework Parameterisierung

## Sicherheitsempfehlungen für Produktion

1. **JWT-Schlüssel:** Verwenden Sie einen starken, zufälligen Schlüssel (min. 256 Bit)
2. **Passwort-Richtlinien:** Implementieren Sie Mindestanforderungen
3. **Rate Limiting:** Begrenzen Sie Login-Versuche
4. **Audit Logging:** Protokollieren Sie sicherheitsrelevante Aktionen
5. **Secrets Management:** Nutzen Sie einen Key Vault für Produktions-Secrets

## Deployment

### Docker

Die Anwendung kann als Docker-Container betrieben werden.

#### Image bauen:

```
docker build -t sauberfix .
```

#### Container starten:

```
docker run -d \
  -p 5000:5000 \
  -e ConnectionStrings__DefaultConnection="Host=db;..." \
  -e JwtSettings__Key="IhrSchluessel" \
  sauberfix
```

### Kubernetes (K3s)

Deployment-Manifeste befinden sich in /k8s/deployment.yaml.

**Voraussetzungen:** 1. K3s Cluster 2. Traefik Ingress Controller 3. cert-manager für TLS 4. GitHub Container Registry Secret

#### Secret für Image Pull erstellen:

```
kubectl create secret docker-registry ghcr-secret \
  --docker-server=ghcr.io \
  --docker-username=<github-user> \
  --docker-password=<github-token>
```

#### Deployment anwenden:

```
kubectl apply -f k8s/deployment.yaml
```

## CI/CD mit GitHub Actions

Bei jedem Push auf main oder master wird automatisch:

1. Docker-Image gebaut
2. Image nach ghcr.io/fabian-lindhardt/sauberfix gepusht

### 3. Tags: latest und Commit-SHA

#### Produktions-Secrets (WICHTIG)

Da das SMTP-Passwort sensibel ist, darf es nicht im Git-Repository gespeichert werden. Es wird über ein Kubernetes Secret in die Anwendung injiziert.

**Secret erstellen:** Vor dem ersten Deployment muss das Secret im Cluster angelegt werden (replace YOUR\_PASSWORD):

```
kubectl create secret generic sauberfix-secrets \
  --from-literal=smtp-password='YOUR_PASSWORD'
```

Das Deployment (k8s/deployment.yaml) liest diesen Wert dann automatisch in die Umgebungsvariable SmtplibSettings\_Password.

---

## Projektstruktur

```
sauberfix/
  Program.vb          # Haupteinstiegspunkt, API-Endpunkte
  Dtos.vb             # Data Transfer Objects
  DatabaseSeeder.vb   # Initiale Datenbefüllung
  Services/
    AuthService.vb    # Authentifizierung
    KundenService.vb  # Kundenverwaltung
    MitarbeiterService.vb # Mitarbeiterverwaltung
    TerminService.vb # Terminverwaltung
    ErinnerungService.vb # Hintergrund-Erinnerungsdienst
  Sauberfix.Data/
    AppDbContext.cs   # EF Core DbContext
    Entities.cs       # Datenbank-Entitäten
    Migrations/       # EF Core Migrationen
  wwwroot/
    index.html        # Hauptanwendung (Tabellen)
    scheduler.html    # Kalenderansicht
  k8s/
    deployment.yaml   # Kubernetes Manifeste
  .github/workflows/
    docker-build.yaml # GitHub Actions CI/CD
  Dockerfile          # Container-Build
  sauberfix.vbproj     # Projektdatei
```

---

## Fehlerbehebung & Technische Herausforderungen

(Projektdokumentation / Pflichtenheftnachweis)

### 1. SMTP E-Mail-Versand

**Problem:** Der E-Mail-Versand schlug Initial fehl (5.7.1 Message does not meet delivery requirements).

**Ursache:** - Port 25 (Standard SMTP) erlaubt oft keine authentifizierte Übermittlung (Submission). - Der Server smtp.flairtec.de verlangt Authentifizierung (AUTH LOGIN), die auf Port 25 deaktiviert war oder blockiert wurde. **Lösung:** - Umstellung auf **Port 587** (STARTTLS). - Implementierung der Authentifizierung mit Benutzername und Passwort via MailKit.



## 2. SSL/TLS Zertifikatsfehler

**Problem:** `SslHandshakeException: The host name (smtp.flairtec.de) did not match the name given in the server's SSL certificate (mx01.flairtec.de).` **Ursache:** Der Mailserver meldet sich mit seinem Hostnamen (mx01), während der Client (smtp) eine andere Domain erwartet. Dies ist bei Shared-Hosting-Umgebungen häufig. **Lösung:** - Deaktivierung der strikten Hostnamen-Prüfung im `ErinnerungsService`:  

```
vb client.CheckCertificateRevocation = False  
client.ServerCertificateValidationCallback = Function(s, c, h, e) True
```

## 3. Zeitzonen-Problematik (Timezones)

**Problem:** Erinnerungs-E-mails wurden nicht versendet, obwohl der Zeitpunkt rechnerisch erreicht war. **Ursache:** - **Datenbank:** Speichert `timestamp without time zone` (Unspecified). - **Benutzer:** Gibt Termin in lokaler Zeit (Deutschland, UTC+1/UTC+2) ein. - **Server/Container:** Läuft in UTC. - **Folge:** Ein Termin um 09:00 wurde als 09:00 UTC interpretiert, obwohl der User 09:00 DE (08:00 UTC) meinte. Der Vergleich `Trigger <= Now` schlug fehl. **Lösung:** - Explizite Umrechnung der aktuellen Serverzeit (`DateTime.UtcNow`) in die Zielzeitzone (`Europe/Berlin`) vor dem Vergleich. - Die Datenbankzeit wird nun als "Deutsche Zeit" interpretiert und mit der "Deutschen Jetzt-Zeit" verglichen.

## 4. Verbindungsprobleme (IPv4/IPv6)

**Problem:** Nach `dotnet run` war die Anwendung unter `localhost:5000` nicht erreichbar ("Connection refused"). **Ursache:** Kestrel bindete standardmäßig nur auf Localhost-Interfaces, was in Container-Umgebungen Probleme bereiten kann. **Lösung:** - Explizites Binding auf 0.0.0.0 in `appsettings.json`:  

```
json "Url": "http://0.0.0.0:5000"
```

 - Dies erlaubt Zugriffe von allen Netzwerkschnittstellen (IPv4).

---

## Lizenz

Dieses Projekt ist urheberrechtlich geschützt.

FullCalendar Scheduler wird unter der Non-Commercial Creative Commons Lizenz verwendet.

---

## Support

Bei Fragen oder Problemen wenden Sie sich an den Projektverantwortlichen.