**ML Gruppe Helpdesk System**

Administrationshandbuch - Erweiterte Edition

*Version 2.0 - 23.10.2025*

Inhalt

[1. Systemübersicht 4](#_Toc212111526)

[Benutzerrollen 4](#_Toc212111527)

[2. Benutzerverwaltung 4](#_Toc212111528)

[Neue Benutzer erstellen 4](#_Toc212111529)

[Passwortverwaltung 4](#_Toc212111530)

[3. Datenbank-Konfiguration 4](#_Toc212111531)

[Datenbank-Optionen 4](#_Toc212111532)

[MySQL/MariaDB Konfiguration 4](#_Toc212111533)

[Installation (Ubuntu/Debian) 4](#_Toc212111534)

[Datenbank und Benutzer erstellen 5](#_Toc212111535)

[Django Konfiguration (settings.py) 5](#_Toc212111536)

[Requirements.txt Update 5](#_Toc212111537)

[PostgreSQL Konfiguration 5](#_Toc212111538)

[Installation (Ubuntu/Debian) 5](#_Toc212111539)

[Datenbank und Benutzer erstellen 5](#_Toc212111540)

[Django Konfiguration (settings.py) 6](#_Toc212111541)

[Requirements.txt Update 6](#_Toc212111542)

[MongoDB Konfiguration (Optional) 6](#_Toc212111543)

[Installation (Ubuntu/Debian) 6](#_Toc212111544)

[Django Konfiguration 6](#_Toc212111545)

[4. Caching und Queue-System 7](#_Toc212111546)

[Redis Installation 7](#_Toc212111547)

[Installation (Ubuntu/Debian) 7](#_Toc212111548)

[Redis Konfiguration überprüfen 7](#_Toc212111549)

[Django Cache mit Redis 7](#_Toc212111550)

[Celery und RabbitMQ 8](#_Toc212111551)

[RabbitMQ Installation 8](#_Toc212111552)

[Celery Installation 8](#_Toc212111553)

[Celery Konfiguration (celery.py) 8](#_Toc212111554)

[Celery Worker starten 8](#_Toc212111555)

[5. Standalone Deployment (Apache/Nginx) 9](#_Toc212111556)

[Apache2 mit Gunicorn 9](#_Toc212111557)

[Installation 9](#_Toc212111558)

[Gunicorn Service erstellen 9](#_Toc212111559)

[Apache VirtualHost konfigurieren 9](#_Toc212111560)

[Nginx mit Gunicorn 10](#_Toc212111561)

[Installation 10](#_Toc212111562)

[Nginx konfigurieren 10](#_Toc212111563)

[6. ISPConfig 3 mit Apache2 12](#_Toc212111564)

[Vorbedingungen 12](#_Toc212111565)

[Schritt 1: Virtuelle Website in ISPConfig erstellen 12](#_Toc212111566)

[Schritt 2: SSH Zugriff aktivieren 12](#_Toc212111567)

[Schritt 3: Applikation hochladen 12](#_Toc212111568)

[Schritt 4: Apache VirtualHost anpassen 12](#_Toc212111569)

[Schritt 5: Gunicorn als Systemd Service 13](#_Toc212111570)

[7. ISPConfig 3 mit Nginx 14](#_Toc212111571)

[Vorbedingungen 14](#_Toc212111572)

[Schritt 1: Virtuelle Website erstellen 14](#_Toc212111573)

[Schritt 2: Nginx Server Block bearbeiten 14](#_Toc212111574)

[Schritt 3: Gunicorn für ISPConfig Nginx 15](#_Toc212111575)

[8. Plesk Integration 16](#_Toc212111576)

[Vorbedingungen 16](#_Toc212111577)

[Schritt 1: Subdomain/Domain erstellen 16](#_Toc212111578)

[Schritt 2: Python Anwendung hochladen 16](#_Toc212111579)

[Schritt 3: Virtual Environment erstellen 16](#_Toc212111580)

[Schritt 4: Node.js Anwendung in Plesk konfigurieren 16](#_Toc212111581)

[Schritt 5: Apache/Nginx Proxy konfigurieren 17](#_Toc212111582)

[9. Knowledge Base Administration 18](#_Toc212111583)

[Artikel erstellen 18](#_Toc212111584)

[10. Support-Levels und Team-Lead 18](#_Toc212111585)

[Support Levels 18](#_Toc212111586)

[11. Statistiken und Berichte 18](#_Toc212111587)

[12. Sicherheit und Datenschutz 18](#_Toc212111588)

[SSL/TLS Zertifikat 18](#_Toc212111589)

[Firewall Konfiguration 18](#_Toc212111590)

[Django Security Settings 19](#_Toc212111591)

[13. Backup und Wartung 19](#_Toc212111592)

[Automatische Datenbank-Backups 19](#_Toc212111593)

[Cron Job für tägliche Backups 19](#_Toc212111594)

[14. Troubleshooting 20](#_Toc212111595)

[Common Issues 20](#_Toc212111596)

[502 Bad Gateway Error 20](#_Toc212111597)

[Datenbankfehler 20](#_Toc212111598)

[Static Files nicht geladen 20](#_Toc212111599)

[Email funktioniert nicht 20](#_Toc212111600)

[Permission denied Fehler 20](#_Toc212111601)

# 1. Systemübersicht

Das ML Gruppe Helpdesk System ist eine Django-basierte Anwendung zur Verwaltung von Support-Tickets. Das System unterstützt mehrere Deployment-Szenarien und Datenbanktypen.

## Benutzerrollen

* Administrator: Vollständiger Zugriff auf alle Funktionen
* Support Agent: Kann Tickets verwalten und Knowledge Base Articles erstellen
* Team Lead (Level 4): Kann Tickets an andere Agents zuweisen
* Customer: Kann nur ihre eigenen Tickets sehen

# 2. Benutzerverwaltung

## Neue Benutzer erstellen

Nur Administratoren können neue Benutzer erstellen. Gehen Sie zum Admin-Panel und navigieren Sie zu "Benutzer" > "Neuer Benutzer".

* Email: Eindeutige Email-Adresse
* Username: Eindeutiger Benutzername
* Vorname und Nachname: Vollständiger Name
* Rolle: Wählen Sie die entsprechende Rolle
* Support Level: Nur für Support Agents (1-4)
* Telefonnummer: Kontaktinformation

## Passwortverwaltung

Neue Kunden erhalten ein initiales Passwort: P@ssw0rd123. Sie müssen das Passwort bei ihrer ersten Anmeldung ändern.

# 3. Datenbank-Konfiguration

## Datenbank-Optionen

Das System unterstützt folgende Datenbanken:

* SQLite - Einfach, keine Installation nötig (Standard)
* MySQL/MariaDB - Produktionsqualität, weit verbreitet
* PostgreSQL - Robust, empfohlen für große Systeme
* MongoDB - NoSQL Alternative (eingeschränkte Unterstützung)

## MySQL/MariaDB Konfiguration

### Installation (Ubuntu/Debian)

sudo apt-get update

sudo apt-get install mysql-server

sudo mysql\_secure\_installation

### Datenbank und Benutzer erstellen

sudo mysql -u root -p

CREATE DATABASE ml\_helpdesk CHARACTER SET utf8mb4 COLLATE utf8mb4\_unicode\_ci;

CREATE USER 'helpdesk'@'localhost' IDENTIFIED BY 'sicheres\_passwort';

GRANT ALL PRIVILEGES ON ml\_helpdesk.\* TO 'helpdesk'@'localhost';

FLUSH PRIVILEGES;

EXIT;

### Django Konfiguration (settings.py)

DATABASES = {  
 'default': {  
 'ENGINE': 'django.db.backends.mysql',  
 'NAME': 'ml\_helpdesk',  
 'USER': 'helpdesk',  
 'PASSWORD': 'sicheres\_passwort',  
 'HOST': 'localhost',  
 'PORT': '3306',  
 'OPTIONS': {  
 'charset': 'utf8mb4',  
 }  
 }  
}

### Requirements.txt Update

pip install mysqlclient

## PostgreSQL Konfiguration

### Installation (Ubuntu/Debian)

sudo apt-get update

sudo apt-get install postgresql postgresql-contrib

### Datenbank und Benutzer erstellen

sudo -u postgres psql

CREATE DATABASE ml\_helpdesk;

CREATE USER helpdesk WITH PASSWORD 'sicheres\_passwort';

ALTER ROLE helpdesk SET client\_encoding TO 'utf8';

ALTER ROLE helpdesk SET default\_transaction\_isolation TO 'read committed';

ALTER ROLE helpdesk SET default\_transaction\_deferrable TO on;

GRANT ALL PRIVILEGES ON DATABASE ml\_helpdesk TO helpdesk;

\q

### Django Konfiguration (settings.py)

DATABASES = {  
 'default': {  
 'ENGINE': 'django.db.backends.postgresql',  
 'NAME': 'ml\_helpdesk',  
 'USER': 'helpdesk',  
 'PASSWORD': 'sicheres\_passwort',  
 'HOST': 'localhost',  
 'PORT': '5432',  
 }  
}

### Requirements.txt Update

pip install psycopg2-binary

## MongoDB Konfiguration (Optional)

### Installation (Ubuntu/Debian)

sudo apt-get update

sudo apt-get install -y mongodb

sudo systemctl start mongodb

### Django Konfiguration

pip install djongo

pip install pymongo

DATABASES = {  
 'default': {  
 'ENGINE': 'djongo',  
 'NAME': 'ml\_helpdesk',  
 'CLIENT': {  
 'host': 'localhost',  
 'port': 27017,  
 }  
 }  
}

# 4. Caching und Queue-System

## Redis Installation

### Installation (Ubuntu/Debian)

sudo apt-get update

sudo apt-get install redis-server

sudo systemctl start redis-server

sudo systemctl enable redis-server

### Redis Konfiguration überprüfen

redis-cli ping

# Sollte PONG zurückgeben

### Django Cache mit Redis

CACHES = {  
 'default': {  
 'BACKEND': 'django\_redis.cache.RedisCache',  
 'LOCATION': 'redis://127.0.0.1:6379/1',  
 'OPTIONS': {  
 'CLIENT\_CLASS': 'django\_redis.client.DefaultClient',  
 }  
 }  
}  
  
pip install django-redis

## Celery und RabbitMQ

### RabbitMQ Installation

sudo apt-get install rabbitmq-server

sudo systemctl start rabbitmq-server

sudo systemctl enable rabbitmq-server

### Celery Installation

pip install celery

pip install celery[redis]

### Celery Konfiguration (celery.py)

import os  
from celery import Celery  
  
os.environ.setdefault('DJANGO\_SETTINGS\_MODULE', 'helpdesk.settings')  
  
app = Celery('helpdesk')  
app.config\_from\_object('django.conf:settings', namespace='CELERY')  
app.autodiscover\_tasks()  
  
# Broker und Backend konfigurieren  
CELERY\_BROKER\_URL = 'redis://localhost:6379/0'  
CELERY\_RESULT\_BACKEND = 'redis://localhost:6379/0'  
CELERY\_ACCEPT\_CONTENT = ['json']  
CELERY\_TASK\_SERIALIZER = 'json'

### Celery Worker starten

celery -A helpdesk worker -l info

# 5. Standalone Deployment (Apache/Nginx)

## Apache2 mit Gunicorn

### Installation

sudo apt-get install apache2

sudo apt-get install python3-pip python3-venv

pip install gunicorn

### Gunicorn Service erstellen

sudo nano /etc/systemd/system/ml-helpdesk.service  
  
[Unit]  
Description=ML Helpdesk Gunicorn Application  
After=network.target  
  
[Service]  
Type=notify  
User=www-data  
Group=www-data  
WorkingDirectory=/var/www/ml-helpdesk  
Environment="PATH=/var/www/ml-helpdesk/.venv/bin"  
ExecStart=/var/www/ml-helpdesk/.venv/bin/gunicorn \  
 --workers 4 \  
 --worker-class sync \  
 --bind unix:/var/www/ml-helpdesk/gunicorn.sock \  
 helpdesk.wsgi:application  
  
[Install]  
WantedBy=multi-user.target  
  
sudo systemctl start ml-helpdesk  
sudo systemctl enable ml-helpdesk

### Apache VirtualHost konfigurieren

sudo nano /etc/apache2/sites-available/ml-helpdesk.conf  
  
<VirtualHost \*:80>  
 ServerName helpdesk.example.com  
 ServerAdmin admin@example.com  
  
 ProxyPreserveHost On  
 ProxyPass / unix:/var/www/ml-helpdesk/gunicorn.sock|http://localhost/  
 ProxyPassReverse / unix:/var/www/ml-helpdesk/gunicorn.sock|http://localhost/  
  
 <Directory /var/www/ml-helpdesk>  
 Require all granted  
 </Directory>  
  
 ErrorLog ${APACHE\_LOG\_DIR}/ml-helpdesk-error.log  
 CustomLog ${APACHE\_LOG\_DIR}/ml-helpdesk-access.log combined  
</VirtualHost>  
  
sudo a2enmod proxy  
sudo a2enmod proxy\_http  
sudo a2ensite ml-helpdesk.conf  
sudo apache2ctl configtest  
sudo systemctl restart apache2

## Nginx mit Gunicorn

### Installation

sudo apt-get install nginx

sudo systemctl start nginx

sudo systemctl enable nginx

### Nginx konfigurieren

sudo nano /etc/nginx/sites-available/ml-helpdesk  
  
upstream ml\_helpdesk {  
 server unix:/var/www/ml-helpdesk/gunicorn.sock fail\_timeout=0;  
}  
  
server {  
 listen 80;  
 server\_name helpdesk.example.com;  
 client\_max\_body\_size 100M;  
  
 location / {  
 proxy\_set\_header Host $host;  
 proxy\_set\_header X-Real-IP $remote\_addr;  
 proxy\_set\_header X-Forwarded-For $proxy\_add\_x\_forwarded\_for;  
 proxy\_set\_header X-Forwarded-Proto $scheme;  
 proxy\_redirect off;  
 proxy\_pass http://ml\_helpdesk;  
 }  
  
 location /static/ {  
 alias /var/www/ml-helpdesk/static/;  
 }  
  
 location /media/ {  
 alias /var/www/ml-helpdesk/media/;  
 }  
  
 error\_log /var/log/nginx/ml-helpdesk-error.log;  
 access\_log /var/log/nginx/ml-helpdesk-access.log;  
}  
  
sudo ln -s /etc/nginx/sites-available/ml-helpdesk /etc/nginx/sites-enabled/  
sudo nginx -t  
sudo systemctl restart nginx

# 6. ISPConfig 3 mit Apache2

## Vorbedingungen

ISPConfig 3 muss installiert sein

Apache2 mit mod\_proxy und mod\_wsgi

Python 3.7+

## Schritt 1: Virtuelle Website in ISPConfig erstellen

1. ISPConfig Control Panel öffnen (https://server.ip:8080)

2. Login mit Administrator-Anmeldedaten

3. "Sites" > "Websites" > "Website hinzufügen"

4. Domain eingeben (z.B. helpdesk.example.com)

5. Document Root: /var/www/clients/client1/web1/web (Standard)

## Schritt 2: SSH Zugriff aktivieren

1. In ISPConfig: "System" > "Benutzer"

2. Benutzer auswählen

3. "SSH/Shell login" auf "jailed shell" oder "nologin" setzen

## Schritt 3: Applikation hochladen

1. FTP/SFTP Zugriff nutzen

2. mini-helpdesk Ordner in Web Root hochladen

3. Berechtigungen: chmod -R 755 /var/www/clients/client1/web1/web/mini-helpdesk

## Schritt 4: Apache VirtualHost anpassen

Editieren Sie folgende Datei:  
/etc/apache2/sites-available/[domain].conf  
  
Fügen Sie hinzu:  
  
<VirtualHost \*:80>  
 ServerName helpdesk.example.com  
 DocumentRoot /var/www/clients/client1/web1/web/mini-helpdesk  
  
 <Directory /var/www/clients/client1/web1/web/mini-helpdesk>  
 AllowOverride All  
 Require all granted  
 </Directory>  
  
 ProxyPreserveHost On  
 ProxyPass /static/ !  
 ProxyPass /media/ !  
 ProxyPass / http://127.0.0.1:8001/  
 ProxyPassReverse / http://127.0.0.1:8001/  
</VirtualHost>

## Schritt 5: Gunicorn als Systemd Service

sudo nano /etc/systemd/system/ml-helpdesk-ispconfig.service  
  
[Unit]  
Description=ML Helpdesk Gunicorn  
After=network.target  
  
[Service]  
Type=notify  
User=web1  
Group=client1  
WorkingDirectory=/var/www/clients/client1/web1/web/mini-helpdesk  
Environment="PATH=/var/www/clients/client1/web1/web/mini-helpdesk/.venv/bin"  
ExecStart=/var/www/clients/client1/web1/web/mini-helpdesk/.venv/bin/gunicorn \  
 --workers 2 \  
 --bind 127.0.0.1:8001 \  
 helpdesk.wsgi:application  
  
[Install]  
WantedBy=multi-user.target  
  
sudo systemctl daemon-reload  
sudo systemctl start ml-helpdesk-ispconfig  
sudo systemctl enable ml-helpdesk-ispconfig

# 7. ISPConfig 3 mit Nginx

## Vorbedingungen

ISPConfig 3 mit Nginx

Python 3.7+

Gunicorn installiert

## Schritt 1: Virtuelle Website erstellen

1. ISPConfig öffnen

2. "Sites" > "Websites" > "Website hinzufügen"

3. Unter "Web Server": Nginx auswählen

4. Domain und Document Root konfigurieren

## Schritt 2: Nginx Server Block bearbeiten

sudo nano /etc/nginx/sites-available/[domain].conf  
  
Beispiel-Konfiguration:  
  
upstream ml\_helpdesk {  
 server 127.0.0.1:8002;  
}  
  
server {  
 listen 80;  
 server\_name helpdesk.example.com;  
 client\_max\_body\_size 100M;  
 root /var/www/clients/client1/web3/web/mini-helpdesk;  
  
 location /static/ {  
 alias /var/www/clients/client1/web3/web/mini-helpdesk/static/;  
 expires 30d;  
 }  
  
 location /media/ {  
 alias /var/www/clients/client1/web3/web/mini-helpdesk/media/;  
 expires 7d;  
 }  
  
 location / {  
 proxy\_pass http://ml\_helpdesk;  
 proxy\_set\_header Host $host;  
 proxy\_set\_header X-Real-IP $remote\_addr;  
 proxy\_set\_header X-Forwarded-For $proxy\_add\_x\_forwarded\_for;  
 proxy\_set\_header X-Forwarded-Proto $scheme;  
 }  
}  
  
sudo systemctl reload nginx

## Schritt 3: Gunicorn für ISPConfig Nginx

sudo nano /etc/systemd/system/ml-helpdesk-nginx.service  
  
[Unit]  
Description=ML Helpdesk Gunicorn for Nginx  
After=network.target  
  
[Service]  
Type=notify  
User=web3  
Group=client1  
WorkingDirectory=/var/www/clients/client1/web3/web/mini-helpdesk  
Environment="PATH=/var/www/clients/client1/web3/web/mini-helpdesk/.venv/bin"  
ExecStart=/var/www/clients/client1/web3/web/mini-helpdesk/.venv/bin/gunicorn \  
 --workers 2 \  
 --bind 127.0.0.1:8002 \  
 helpdesk.wsgi:application  
  
[Install]  
WantedBy=multi-user.target  
  
sudo systemctl daemon-reload  
sudo systemctl start ml-helpdesk-nginx  
sudo systemctl enable ml-helpdesk-nginx

# 8. Plesk Integration

## Vorbedingungen

Plesk 17.8+ installiert

Python 3.7+ aktiviert

Node.js Modul installiert (optional)

## Schritt 1: Subdomain/Domain erstellen

1. Plesk Control Panel öffnen

2. "Domains" > Domain auswählen

3. "Subdomains" > "Add Subdomain"

4. Subdomain Name eingeben (z.B. "helpdesk")

5. Bestätigen

## Schritt 2: Python Anwendung hochladen

1. FTP/SFTP nutzen oder Files Manager in Plesk  
2. mini-helpdesk Ordner hochladen in: /httpdocs/helpdesk/  
3. Dateiberechtigungen setzen: 755 für Ordner, 644 für Dateien

## Schritt 3: Virtual Environment erstellen

SSH als root:  
  
cd /var/www/vhosts/example.com/httpdocs/helpdesk  
python3 -m venv .venv  
source .venv/bin/activate  
pip install -r requirements.txt  
pip install gunicorn

## Schritt 4: Node.js Anwendung in Plesk konfigurieren

1. In Plesk: "Domains" > Domain > "Node.js"  
2. "Add Node.js Application" klicken  
3. Einstellungen:  
 - Path: /httpdocs/helpdesk/  
 - Document root: /httpdocs/helpdesk/public  
 - Application mode: production  
 - Port: 3000 (oder beliebiger Port)  
 - NPM Scripts: verwende gunicorn statt npm  
  
4. Alternativ: Custom Startup Script  
 Script: /var/www/vhosts/example.com/.venv/bin/gunicorn \  
 --workers 2 --bind 127.0.0.1:3000 helpdesk.wsgi:application

## Schritt 5: Apache/Nginx Proxy konfigurieren

Plesk erstellt automatisch Proxy-Regeln. Falls nicht:  
  
Apache (httpd.conf oder htaccess):  
RewriteEngine On  
RewriteRule ^(.\*)$ http://127.0.0.1:3000/$1 [P,L]  
  
Nginx (wird in Plesk-Interface konfiguriert)

# 9. Knowledge Base Administration

## Artikel erstellen

Support Agents und Administratoren können Knowledge Base Artikel erstellen. Artikel sollten häufige Probleme und deren Lösungen dokumentieren.

* Titel: Aussagekräftiger Titel
* Kategorie: Thematische Zuordnung
* Inhalt: Detaillierte Anleitung (mit RichText-Editor)
* Tags: Für bessere Auffindbarkeit
* Veröffentlicht: Status des Artikels

# 10. Support-Levels und Team-Lead

## Support Levels

Es gibt 4 Support Levels für Agents:

* Level 1: Junior Support (kann nur einfache Tickets lösen)
* Level 2: Senior Support (normale Tickets)
* Level 3: Expert Support (komplexe Probleme)
* Level 4: Team Lead (kann Tickets zuweisen und delegieren)

# 11. Statistiken und Berichte

Das System bietet umfangreiche Statistiken zur Überwachung der Support-Qualität. Nur Administratoren und Team Leads haben Zugriff auf Statistiken.

# 12. Sicherheit und Datenschutz

## SSL/TLS Zertifikat

Verwenden Sie Let's Encrypt für kostenlose Zertifikate:

sudo apt-get install certbot python3-certbot-apache

sudo certbot certonly --apache -d helpdesk.example.com

## Firewall Konfiguration

sudo ufw allow 22/tcp

sudo ufw allow 80/tcp

sudo ufw allow 443/tcp

sudo ufw enable

## Django Security Settings

# settings.py  
  
DEBUG = False  
ALLOWED\_HOSTS = ['helpdesk.example.com']  
  
# HTTPS  
SECURE\_SSL\_REDIRECT = True  
SESSION\_COOKIE\_SECURE = True  
CSRF\_COOKIE\_SECURE = True  
SECURE\_BROWSER\_XSS\_FILTER = True  
SECURE\_CONTENT\_SECURITY\_POLICY = {  
 'default-src': ("'self'",),  
}

# 13. Backup und Wartung

## Automatische Datenbank-Backups

#!/bin/bash

BACKUP\_DIR="/backups/ml-helpdesk"

DATE=$(date +%Y%m%d\_%H%M%S)

# PostgreSQL Backup

pg\_dump ml\_helpdesk > $BACKUP\_DIR/db\_backup\_$DATE.sql

gzip $BACKUP\_DIR/db\_backup\_$DATE.sql

# MySQL Backup

mysqldump -u helpdesk -p ml\_helpdesk > $BACKUP\_DIR/db\_backup\_$DATE.sql

## Cron Job für tägliche Backups

crontab -e

0 2 \* \* \* /usr/local/bin/backup-helpdesk.sh

# 14. Troubleshooting

## Common Issues

### 502 Bad Gateway Error

Gunicorn nicht erreichbar. Überprüfen Sie: systemctl status ml-helpdesk

### Datenbankfehler

Überprüfen Sie Datenbank-Verbindung und Credentials in settings.py

### Static Files nicht geladen

Führen Sie aus: python manage.py collectstatic

### Email funktioniert nicht

Überprüfen Sie SMTP-Einstellungen und Firewall-Regeln

### Permission denied Fehler

Überprüfen Sie Dateiberechtigungen: chmod -R 755 /var/www/ml-helpdesk

*© 2025 ML Gruppe Helpdesk System - Erweiterte Administrationshandbuch*