Kimai Zeiterfassung

Installtion

Installation via Docker

Beschreibung

Kimai via Docker mit https via Letsencrypt.

Installation

Vorbereitung

Sollte Apparmor enabled sein, wie das bei hetzner der Fall ist dann noch eben die Apparmor Pakete installieren

```
apt-get install apparmor apparmor-utils curl docker.io docker-compose
```

Als erstes erstellen wir unsere Verzeichnisse

```
mkdir -p /root/kimai/mariadb
mkdir -p /root/kimai/data
mkdir -p /root/kimai/plugins
```

Nun erstellen wir im Verzeichnis /root/kimai eine neue Datei namens .env

```
nano /root/kimai/.env
```

Inhalt:

```
# MySQL Credentials
MYSQL_ROOT_PASSWORD=12345678
MYSQL_USER=kimai
MYSQL_PASSWORD=1234567890
MYSQL_DATABASE=kimai

# Volume directories
```

```
MARIADB_VOLUME_DIR=/root/kimai/mariadb
DATA_VOLUME_DIR=/root/kimai/data
```

Docker-compose Datei erstellen

```
nano /root/kimai/docker-compose.yml
```

Inhalt

```
ersion: '3.3'

services:
  mariadb:
    image: mariadb:latest
    restart: unless-stopped
    environment:
      MYSQL_ROOT_PASSWORD: ${MYSQL_ROOT_PASSWORD}
      MYSQL_USER: ${MYSQL_USER}
      MYSQL_PASSWORD: ${MYSQL_PASSWORD}
      MYSQL_DATABASE: ${MYSQL_DATABASE}
    volumes:
      - ${MARIADB_VOLUME_DIR}:/var/lib/mysql

  kimai:
    image: kimai/kimai2:apache
    restart: unless-stopped
    environment:
      - DATABASE_URL=mysql://${MYSQL_USER}:${MYSQL_PASSWORD}@mariadb/${MYSQL_DATABASE}
    volumes:
      - ${DATA_VOLUME_DIR}:/opt/kimai/var
      - ./plugins:/opt/kimai/var/plugins

  caddy:
    image: caddy:latest
    container_name: caddy-reverse-proxy
    ports:
      - "80:80"
      - "443:443"
```

```
    volumes:
      - ./Caddyfile:/etc/caddy/Caddyfile
      - ./caddy_data:/data
      - ./caddy_config:/config
    restart: unless-stopped
```

Nun Caddyfile erstellen

```
nano /root/kimai/Caddyfile
```

Inhalt

```
zeit.example.tld {
    reverse_proxy http://kimai:8001
    tls info@example.tld {
        #ca https://acme-staging-v02.api.letsencrypt.org/directory
    }
}
```

Nun sicher stellen das die ports 80 und 443 auch über die Domain auf den Server erreichbar sind.

hier ein kleines Script das auf den port 80 lauscht um dies zu testen

```
nano /root/kimai/testeweb.sh
```

Inhalt

```
#!/bin/bash

# Funktion, die aufgerufen wird, wenn das Skript ein SIGINT-Signal (Ctrl+C) empfängt
cleanup() {
    echo "Server wird beendet..."
    exit 0
}

# Trap für das SIGINT-Signal einrichten
trap cleanup SIGINT
```

```
# Endlosschleife für den Server
while true; do
    # Listen on port 80
    { echo -ne "HTTP/1.1 200 OK\r\nContent-Length: $(echo -n "Hello, World!" | wc -c)\r\n\r\nHello, World!"; } | nc
-l -p 80 -q 1
done
```

Danach ausführbar machen

```
chmod +x /root/kimai/testeweb.sh
```

nun das Script starten, kann mit strg+c wieder abgebrochen werden.
Nun kann getestet werden ob Firewall/Portforwarding richtig funktioniert, idem in einem Webbrowser die public ip aufegrufen wird.
Es muss eine hello World Seite zurück gegben werden.

Container Starten

Ins kimai Verzeichnis gehen

```
cd /root/kimai
```

Nun den Conatiner starten

```
 docker-compose up -d
```

Benutzer anlegen

Die Rollen

Role name	Description
User	Normal user can track their working times, see basic reports and change their own preferences. Technical name: `ROLE_USER`
Teamlead	Manages teams with permissions for invoices and access to all **team** timesheets. Technical name: `ROLE_TEAMLEAD`

Role name	Description
Administrator	Can manage all content and timesheet related data, but lack user administration and system privileges. Technical name: `ROLE_ADMIN`
System-Admin	Has permissions to manage everything in Kimai, from content to timesheets to users, plugins and system configurations. Technical name: `ROLE_SUPER_ADMIN`

die container id mit docker ps herausfinden

docker exec -ti <containerid> \
    /opt/kimai/bin/console kimai:user:create <username> <emailadresse> ROLE_SUPER_ADMIN

Beispiel:

docker exec -ti 32da986c57c2 \
    /opt/kimai/bin/console kimai:user:create admin admin@example.com ROLE_SUPER_ADMIN

Dann kann ein Kennwort vergeben werden. Muss mindestens 8 Zeichen haben

Benutzer Passwort ändern

die container id mit docker ps herausfinden

docker exec -ti <containerid> \
    /opt/kimai/bin/console kimai:user:password <username>

Beispiel:

docker exec -ti 32da986c57c2 \
    /opt/kimai/bin/console kimai:user:password admin

Dann kann ein neues Kennwort vergeben werden. Muss mindestens 8 Zeichen haben

Mobile App

Beschreibung:

Für Kimai gibt es auch eine Mobile App.

Aus dem AppStore die Mobile App installieren:

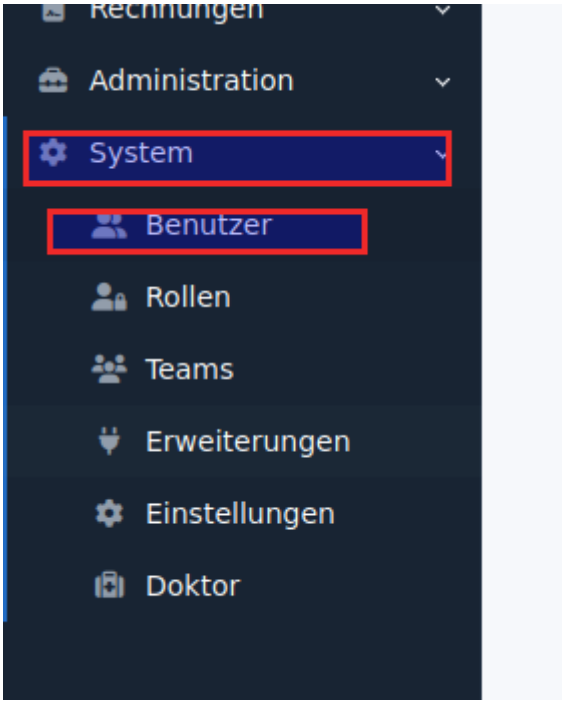
iOS link zum Store

Android Link zum Store

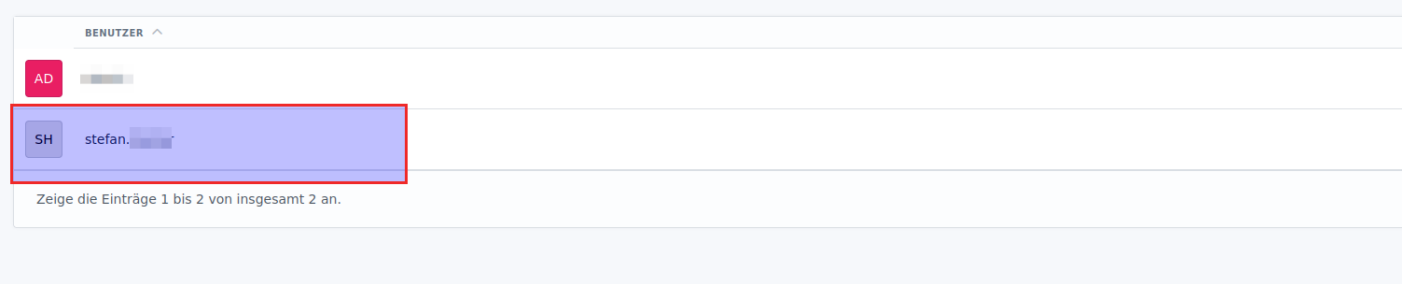
Benutzer freischalten im Kimai Backend.

Damit sich der Benutzer anmelden kann, muss in den Benutzereinstellungen ein APi Passwort (Nicht Token) hinterlegt werden.
Beim Passwort kommt der Hinweis das diese methode veraltet ist, aber die Mobil App nutzt zur zeit nur Passwort.

Dazu gehen wir Auf System -> Benutzer

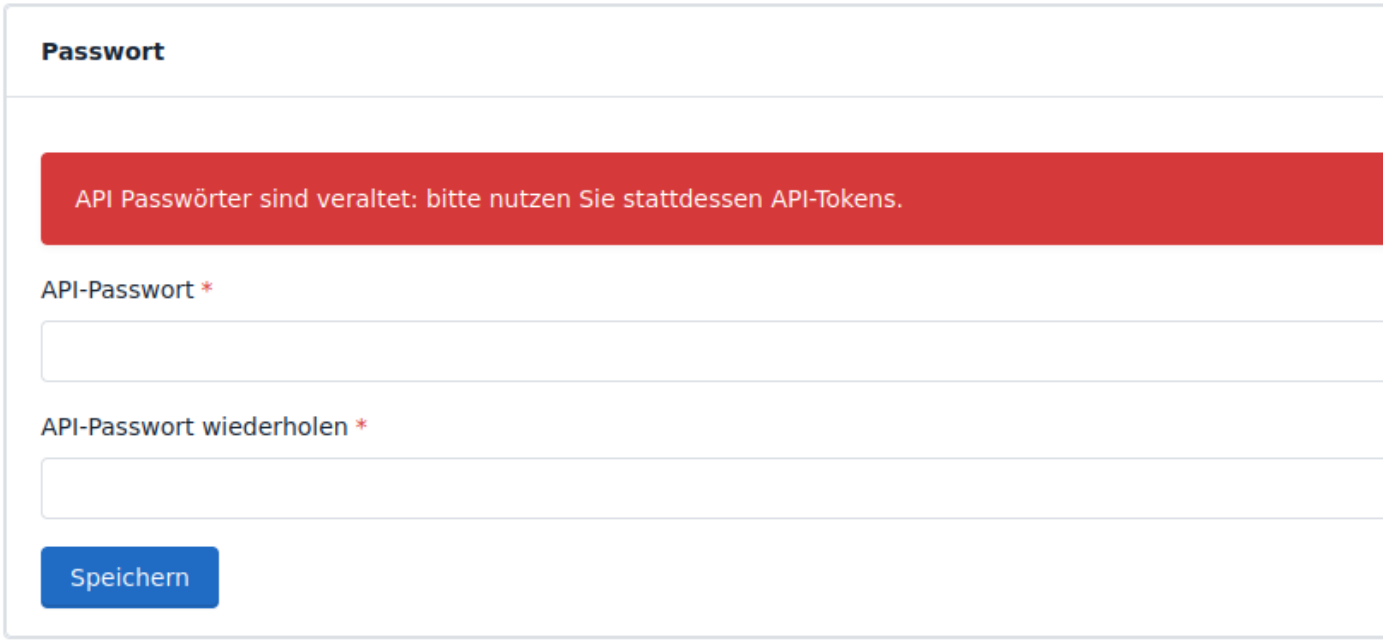


Dann den benutzer anklicken, für den wir das API Passwort erstellen wollen



Dort dann oben auf bearbeiten klicken und API Zugang auswählen

Nun das neue Kennwort setzten und speichern.



Einloggen auf der Mobile App

Create Workspace anklicken.

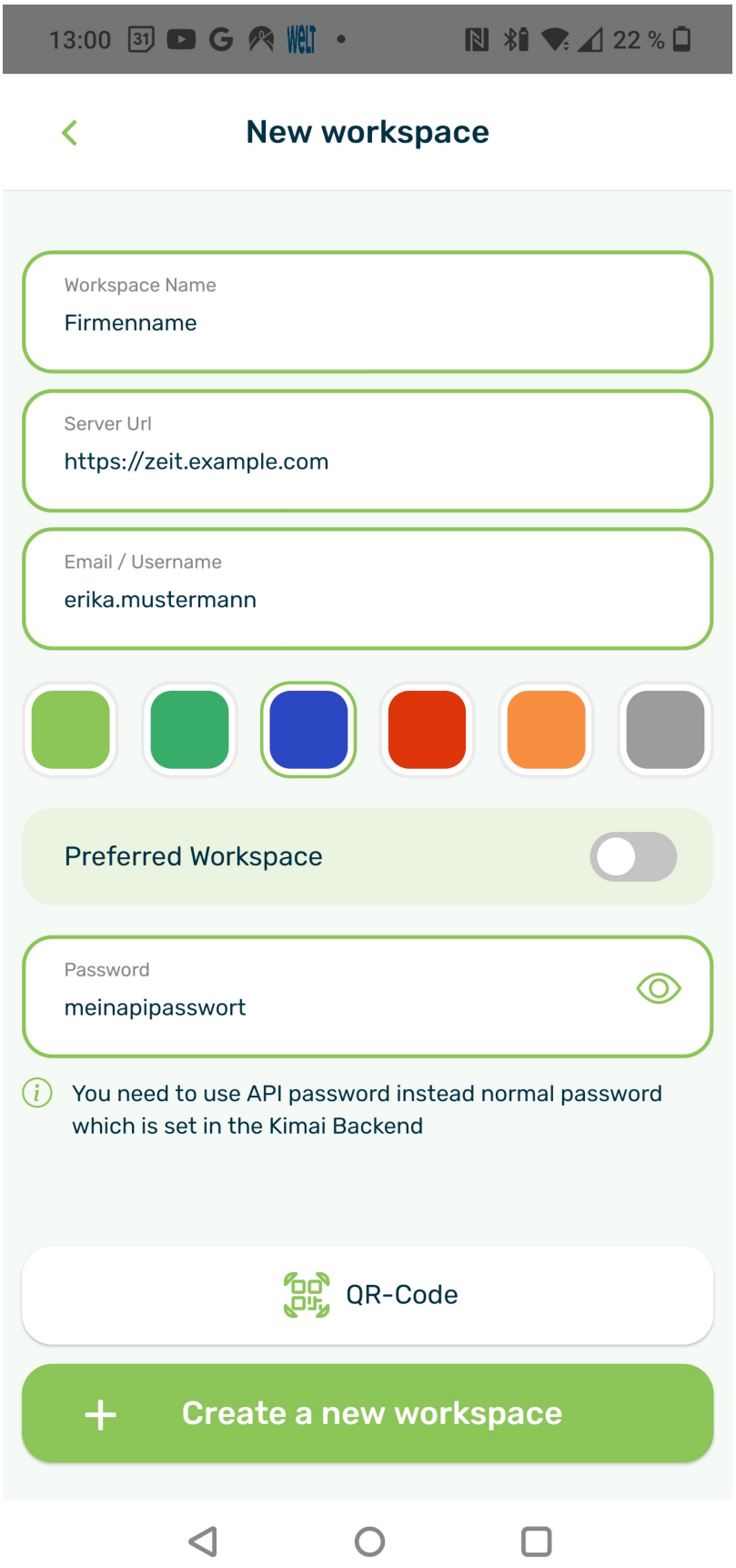
Daten ausfüllen.

Workspace Name, zum Beispiel Firmenname.

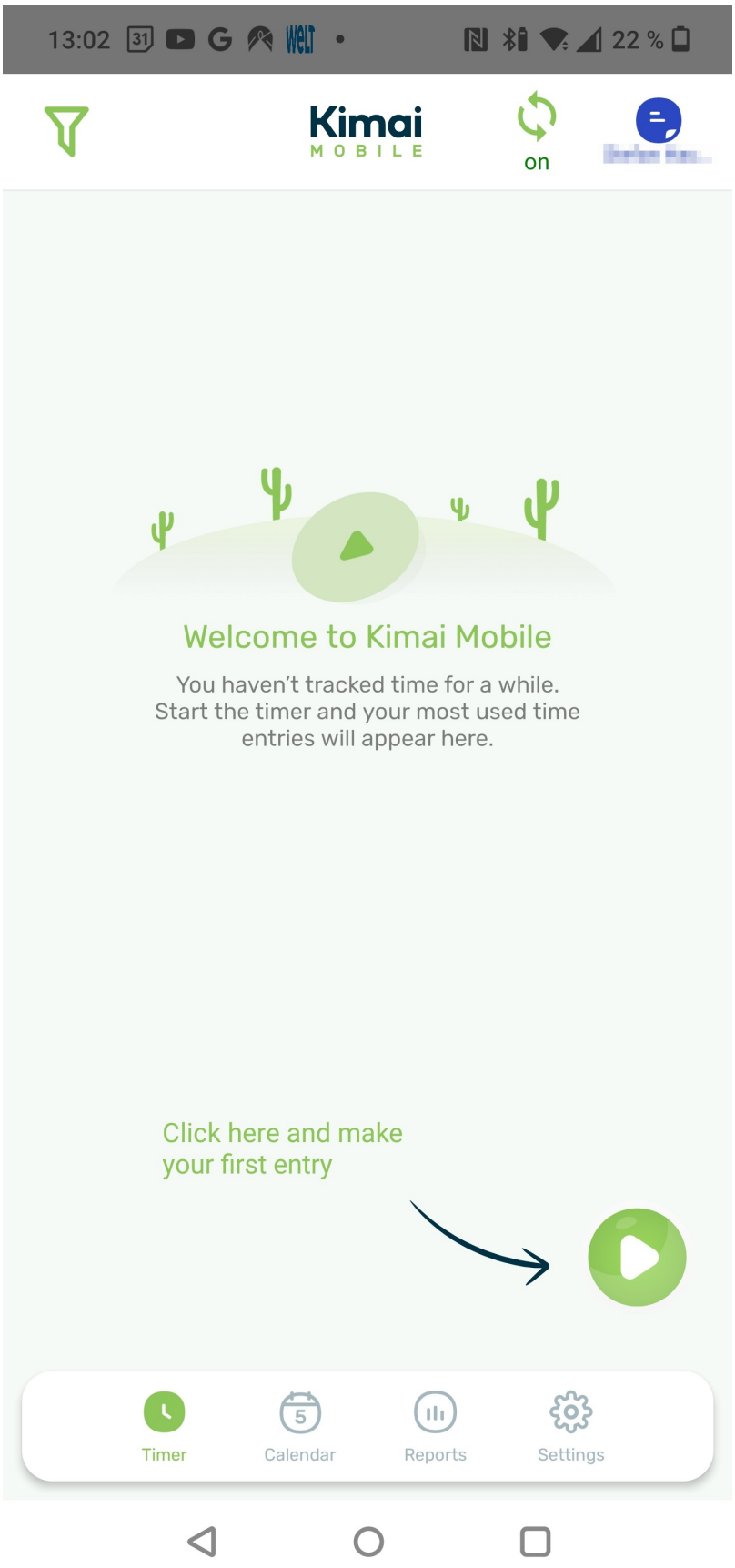
Server URL : die Kimai Seite https://vorweg schreiben
Die APP Funktioniert nur über https

Email / Username : den Benutzernamen eintragen
Password: Das API Kennwort

Nun auf Create a new Workspace klicken.



Und eingeloggt:


13:02
22 %
Kimai
MOBILE
on
Welcome to Kimai Mobile
You haven't tracked time for a while.
Start the timer and your most used time
entries will appear here.
Click here and make
your first entry
Timer
Calendar
Reports
Settings

Kimai Terminal

Einleitung und Teile Liste

Beschreibung:

Ein KIMAi terminal Kostengüntig mit nem SONOFF Switch.
Günstiger bekommt man es nicht zusammen.

Teile Liste:

SONOFF Switch (SONOFFT4EU1C, t4eu1c) SonOff Tooch Store Link

Wenn man kein SONOFF kaufen möchte kann man auch einfach den Microcontroller nutzen der WLAN schon drauf hat:

AZDelivery-NodeMCU-ESP8266-ESP-12E-Development
ich find den SONOFF aber Praktisch, alles schon drin in einem Gehäuse

RFID Plate mit 5 Dongle : 5er Pack Rabatt (Sonst kostent einer schon 7 EURO ohne Dongle) RFID-Kit-Arduino-Raspberry-SPI-Schnittstelle

OLDED Display
(APKLVSR 3PCS 0,96 Zoll OLED Display I2C 128 x 64 Pixel I2C Display Modul, (Gelb und Blau)) :
Link : APKLVSR-Display-kompatibel-Arduino-Raspberry

UART Interface (Wird nur für SONOFF gebraucht, das NodeMCU hat einen USB Anschluss : UART-Wandler-Adapter-serielle-Schnittstelle-Converter-Anschlussleitung

100 Dongle zum Nachbestellen extrem Klasse Preis : Chips-125KHz-Transponder 100 Stück

Plugin Invoice Bundle

Installation invoiceBundle

Beschreibung:

Ein Plugin mit dem Gleichzeitig Stundenzettel generiert und Rechnungen erstellt werden können.
Mich Persönlich interessieren nur die Stundenzettel

Da ich Kimai im Docker betreibe ist die Installation an dem Docker setup Angelehnt

Vorrausetzungen:

```
apt install unzip
```

Installation bei Kimai Docker:

In Das Verzeichnis Plugins die Zip Datei exhtrahieren. InvoiceBundle-2.2.0.zip
Dazu per scp die Datei auf den Server kopieren, versionsnummer kann natürlich varieren

```
scp InvoiceBundle-2.2.0.zip root@zeit.example.tld:/root/kimai
```

Nun entpacken

```
cd root/kimai
unzip InvoiceBundle-2.2.0.zip
```

ls Ausgabe:

Nun haben Wir ein verzeichnis InvoiceBundle-2.2.0

```
ls
caddy_config  caddy_data  Caddyfile  certbot-etc  certbot-log  certbot-www  data  docker-compose.yml
InvoiceBundle-2.2.0  InvoiceBundle-2.2.0.zip  mariadb  plugins
```

Dieses umbenennen und verschieben

```
mv InvoiceBundle-2.2.0 plugins/InvoiceBundle
```

Nun muss im InvoiceBundle Verzeichnis der inhsalt so aussehen

ls Ausgabe:

```
ls
CHANGELOG.md  composer.json  Controller  DependencyInjection  EventSubscriber  Form  Invoice
InvoiceBundle.php  LICENSE  phpstan.neon README.md  Resources  Tests
```

Nun nur noch den Cache leeren, dazu im kima Container einloggen

```
docker-compose exec kimai /opt/kimai/bin/console kimai:reload --env=prod
```

Ausgabe bei mir mit Fehler zu alt. :

```
root@debian-kimai-4gb-nbg1-2:~/kimai# docker-compose exec kimai /opt/kimai/bin/console kimai:reload --
env=prod

In Kernel.php line 115:

  Bundle "InvoiceBundle" requires minimum Kimai version 22700, but yours is lower: 2.18.0 (21800). Please
update Kimai or use a lower Plugin version.
```

Also kimai aktualiseren

```
docker pull kimai/kimai2:apache
docker-compose up -d
docker-compose exec kimai /opt/kimai/bin/console kimai:reload --env=prod
docker-compose down
docker-compose up -d
docker-compose exec kimai /opt/kimai/bin/console assets:install
docker-compose exec kimai /opt/kimai/bin/console kimai:reload --env=prod
```

Nun Verzeichnis Berechtigungen setzten, Dazu in den Container einloggen

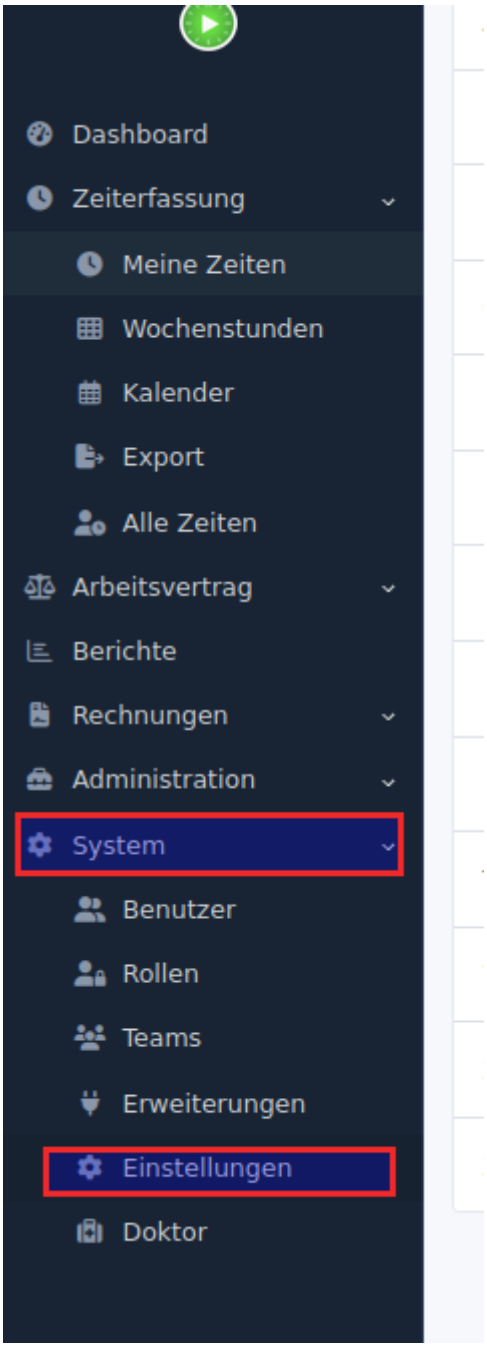
```
docker-compose exec kimai /bin/bash
```

Nun ausführen

```
cd /opt/kimai/var
chown -R :www-data .
chmod -R g+r .
chmod -R g+rw /opt/kimai/var/
```

Nun Einstellungen in Kimai:

Unter System -> Einstellungen



Nun runter scrollen bis Invoice
Dort Rechungsnummerformat Sprache und template für den Stundenzettel Rechung einstellen

Rechnungen

Rechnungsnummer-Format *

{Y}/{cy,3}

Erlaubte Ersetzer: {Y}, {y}, {M}, {m}, {D}, {d}, {date}, {cc}, {ccy}, {ccm}, {ccd}, {cu}, {cuy}, {cum}, {cud}, {ustaff}, {uid}, {c}, {cy}, {cm}, {cd}, {cname}, {cnumber}

Sprache für Formatierungen

Deutsch

Die gewählte Sprache wird genutzt um Datums-, Zeit- und Geldwerte zu formatieren. Die Sprache der Rechnungsvorlage wird danach nur noch für Übersetzungen verwendet.

Export Template

timesheet.pdf.twig

Das hier angegebene Template wird für die Generierung des Stundenzettel-PDFs für neue Rechnungen verwendet.

Speichern Zurücksetzen

Einrichtung Invoice Bundle

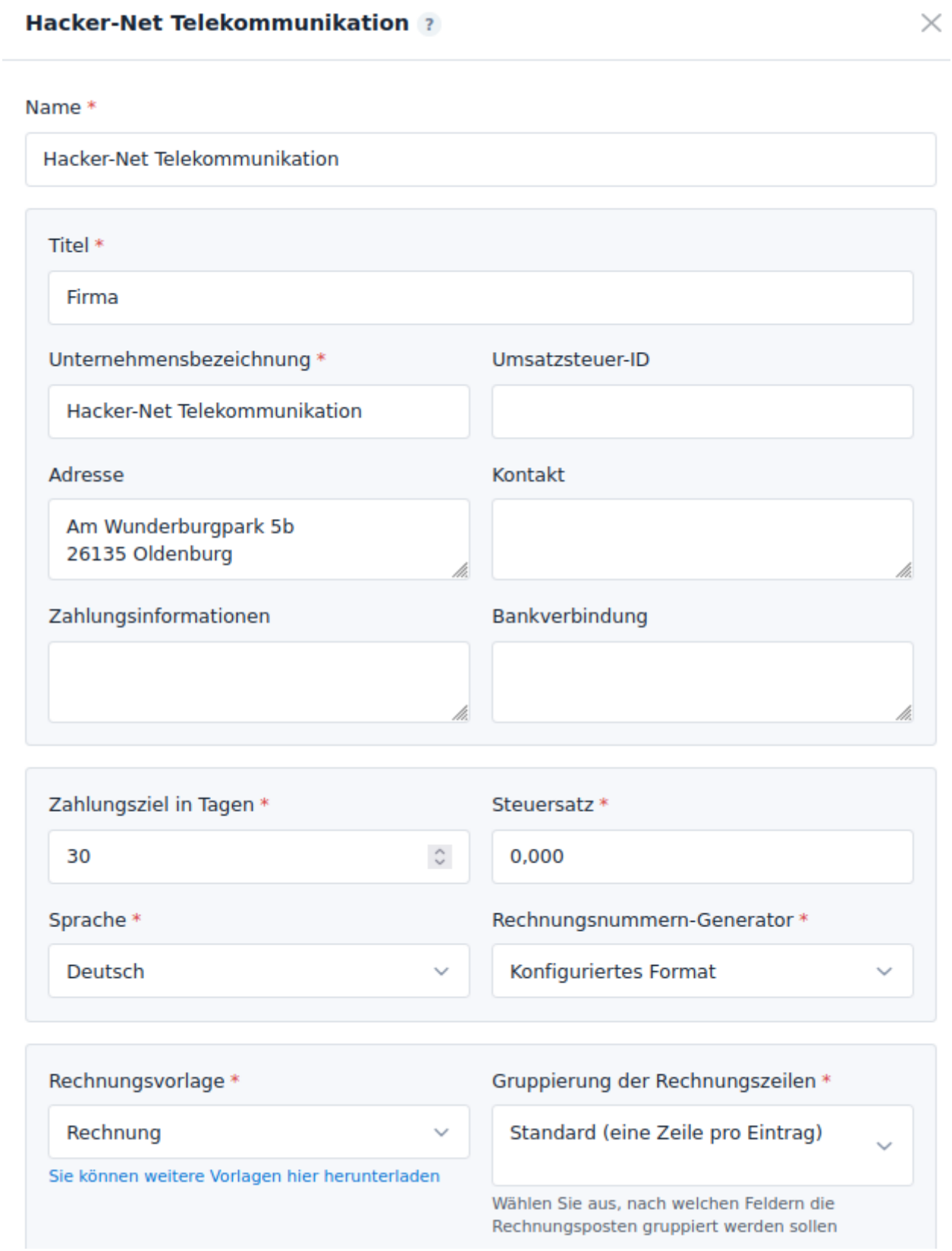
Beschreibung:

nach dem das Plugin installiert.
Das format eingerichtet ist.

geht nun weiter

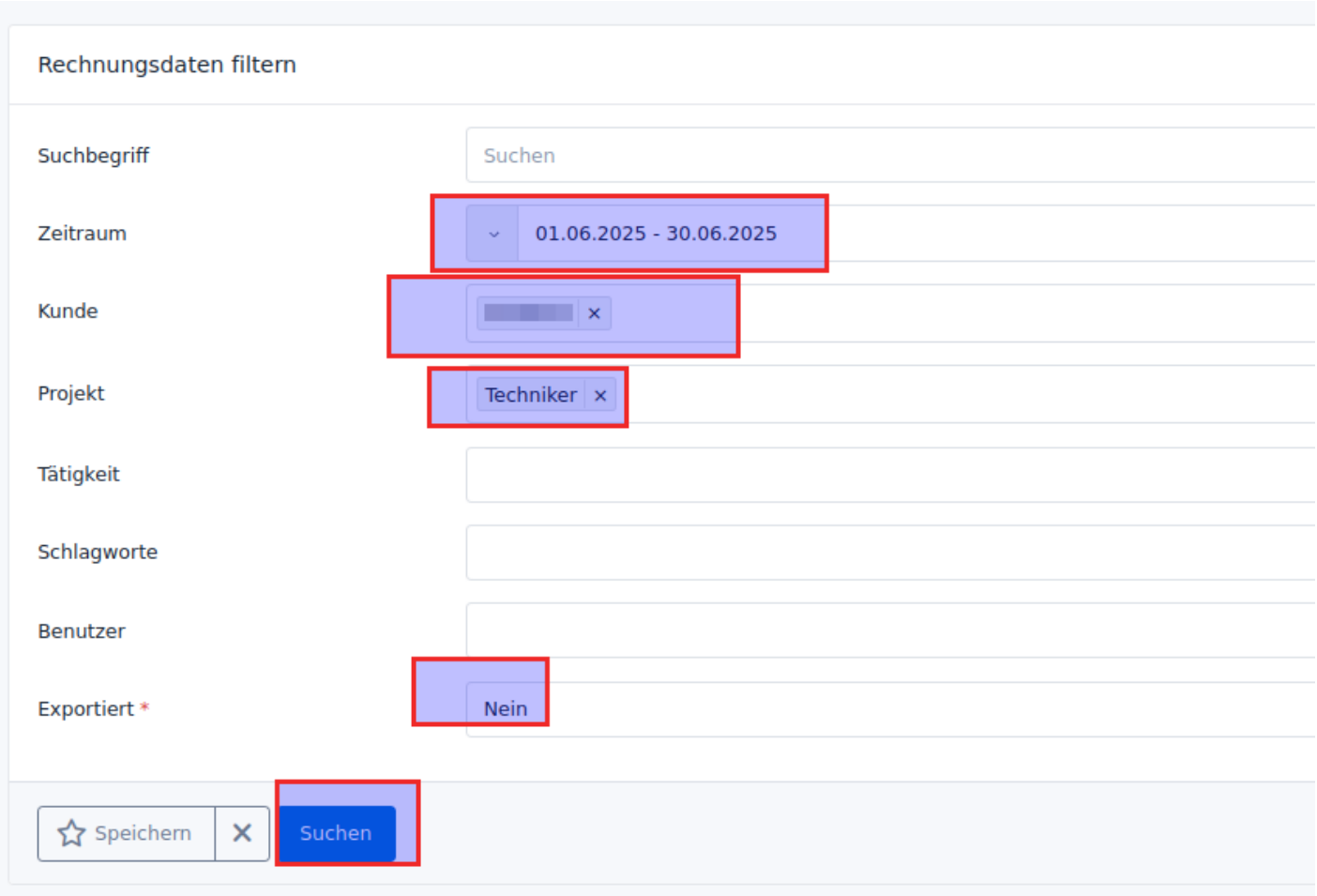
Eigene Firmenvorlage einrichten:

Unter Rechnungen -> Rechungsvorlagen -> Auf erstellen klicken

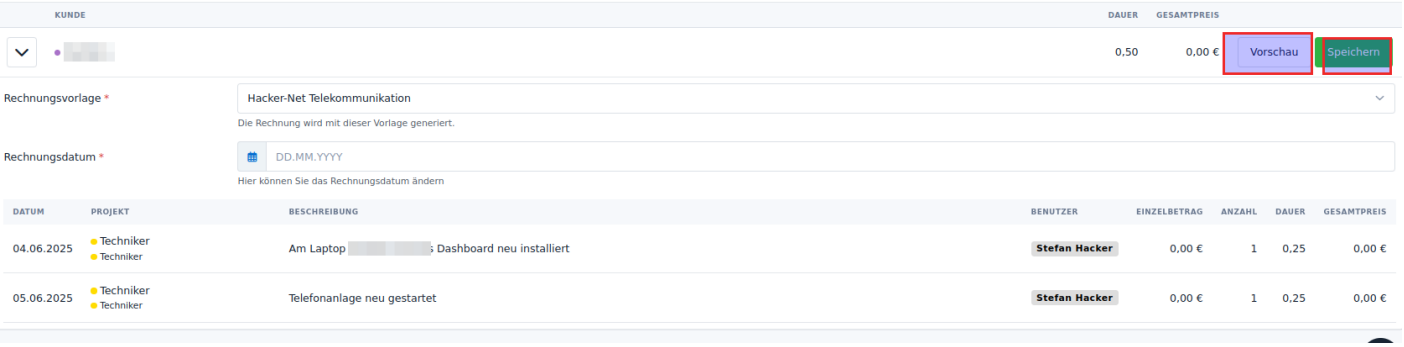


Rechnung / Stundenzettel erstellen:

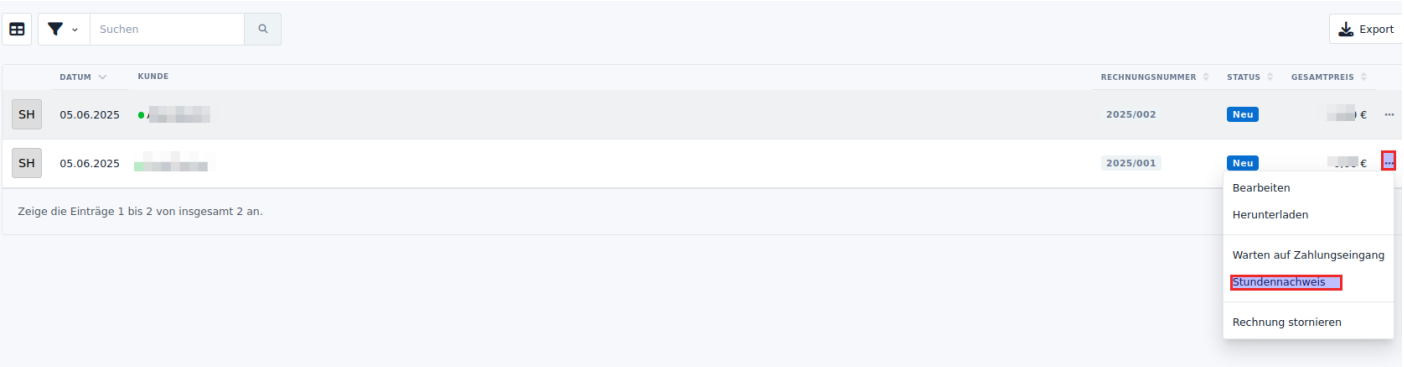
Nun Auf Rechnungen -> Rechnung erstellen klicken und Zeitraum Kunde Projekt auswählen. Will man alle projekte, das Projekt wieder entfernen.
Exportier nein und auf suchen klicken



Nun erscheint eine Liste mit den gefunden Einträgen.
Durch klick auf Speichern werden alle Eintrage auf eien Rechung / Leistungsnachweis gepackt.
Möchte man das getrennt haben. Einfach nach mehren Datumsbereichen suchen und mehrere Rechungen abspeichern.
Über die Buttons Vorschau kann man sich die Rechnung anschauen und speichern dann absenden.



Nun kann Unter Rechnungen -> rechnungshistorie dier Stundenzettel downgeloaded werden


Suchen
Export
DATUM
KUNDE
RECHNUNGSNUMMER
STATUS
GESAMTPREIS
SH
05.06.2025
2025/002
Neu
SH
05.06.2025
2025/001
Neu
Zeige die Einträge 1 bis 2 von insgesamt 2 an.
Bearbeiten
Herunterladen
Warten auf Zahlungseingang
Stundennachweis
Rechnung stornieren

Fertig