**Benutzer für** [**http://172.22.100.66/**](http://172.22.100.66/)**:**

**SSH/SCP:** XXXXX:XXXXX   
 **Webseite:** XXXXX:XXXXX

*(auf Anfrage bei Pascal Blum, ansonsten VM neu aufsetzen)*

**Virtual Machine:**

\* OS: Linux Alpine  
\* 384 MB RAM, 5 GB Storage  
\* Nginx  
\* Gunicorn  
\* Python 3.12 oder höher

**Installation:**

Linux Alpine nach Anweisung installieren.  
Locale auf *gmt+1/de* setzen.

=> apk add tzdata  
=> ln -sf /usr/share/zoneinfo/Europe/Berlin /etc/localtime  
=> apk add --no-cache musl-locales musl-locales-lang  
=> nano /etc/profile

Folgende Zeile hinzufügen: export LANG=de\_DE.UTF-8

**Das Python Skript umfasst folgende Dateien/Ordner:**

/data/ => Die CSV Dateien mit den Daten, die entweder mit einem Texteditor/Excel bearbeitet werden können, oder über die Admin Oberfläche.  
/templates/ => Die HTML Templates für die Oberfläche  
/static/ => Die JS/CSS und generierte Dateien für die Oberfläche  
/exporters/ => Module für generate\_plan.py für alternative Formate z.B. ICS  
/app.py => Das hauptsächliche Python Skript das den Server startet  
/generate\_plan.py => Das Python Skript welches den Plan generiert (\_legacy\_main.py) ist das alte CLI Interface und wird nicht benötigt für das Web-Frontend)

Alle genannten Verzeichnisse und Dateien auf der linken Seite müssen anschließend auf den Server gesetzt werden. In meinem Fall habe ich mich für /opt/webservice/ entschieden. Dann die requirements installieren mit:

=> pip install -r /opt/webservice/requirements.txt

**Gunicorn Service:**

Gunicorn wie üblich installieren und dann einen Service erstellen der /opt/webservice/app.py über port 666 laufen lässt, bzw. je nach Ort/Port.

**Nginx:**

Eine Config erstellen für Nginx die das .htaccess und die HTTP/HTTPS Anfragen zum Python Flask Server weiterleitet.

=> touch /etc/nginx/http.d/<spüliplan server name>.conf  
 => nano /etc/nginx/http.d/<spüliplan server name>.conf  
  
Folgender Inhalt soll eingefügt/ergänzt werden für Nginx:

|  |
| --- |
| # Handle HTTP requests (port 80)  server {  listen 80;  server\_name \_;  location / {  proxy\_pass http://127.0.0.1:666;  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;  }  # Protect /admin/  location /admin/ {  auth\_basic "Restricted Access";  auth\_basic\_user\_file /etc/nginx/.htpasswd;  proxy\_pass http://127.0.0.1:666; # Your Flask app  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;  }  # Protect /api/  location /api/ {  auth\_basic "Restricted Access";  auth\_basic\_user\_file /etc/nginx/.htpasswd;  proxy\_pass http://127.0.0.1:666;  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 $scheme;  }  }  # Handle HTTPS requests (port 443)  server {  listen 443 ssl;  server\_name \_;  # Use self-signed certificate or a real one if you have one  ssl\_certificate /etc/nginx/ssl/nginx.crt;  ssl\_certificate\_key /etc/nginx/ssl/nginx.key;  # Redirect all HTTPS traffic to HTTP  location / {  return 301 http://$host$request\_uri;  }  # Protect /admin/  location /admin/ {  auth\_basic "Restricted Access";  auth\_basic\_user\_file /etc/nginx/.htpasswd;  proxy\_pass http://127.0.0.1:666;  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;  }  # Protect /api/  location /api/ {  auth\_basic "Restricted Access";  auth\_basic\_user\_file /etc/nginx/.htpasswd;  proxy\_pass http://127.0.0.1:666;  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;  }  } |

Beachte das dies davon ausgeht das der Gunicorn/Python Server über port 666 läuft und die .htpasswd in /etc/nginx/ generiert wird für die Passwort Abfrage.