Installation

- Installation über einen Docker container mit ssl

Installation über einen Docker container mit ssl

Voraussetzungen:

Docker: https://www.docker.com/get-started

Docker installieren

Siehe Buch -> Installation

Budibase installieren

Als Befehl ohne ssl einfach port 10000

```
sudo docker run -d -t \
  --name=budibase \
  -p 10000:80 \
  -v /home/duffy/budibasedata:/data \
  --restart unless-stopped \
  budibase/budibase:latest
```

Environment Parameter

Variablenname	Beschreibung
CUSTOM_DOMAIN	Wenn eine Domain im Format domain.com festgelegt ist, versucht Budibase automatisch, ein SSL-Zertifikat zu erstellen und HTTPS für diese Domain zu verwenden. Die Domain muss bereits auf den Budibase-Container gerichtet sein, damit dies korrekt überprüft werden kann.
INTERNAL_API_KEY	Ein API-Schlüssel, der zum Zugriff auf viele Kernkomponenten verwendet werden kann. Dieser sollte auf eine zufällige Zeichenkette aktualisiert werden.
JWT_SECRET	Ein geheimer Schlüssel, der zum Sichern aller Sitzungen mit Budibase verwendet wird. Dies sollte auf eine zufällige Zeichenkette aktualisiert werden. Beachten Sie, dass das Ändern dieser Zeichenkette alle vorhandenen Sitzungen ungültig macht.

Variablenname	Beschreibung
MINIO_ACCESS_KEY und MINIO_SECRET_KEY	Diese beiden Umgebungsvariablen sollten auf eine Kombination aus zufälligen Zeichenketten festgelegt werden, um den Zugriff auf MinIO abzusichern. Diese können auch verwendet werden, um sich im MinIO-Browser anzumelden, wenn gewünscht.
REDIS_PASSWORD	Das Passwort, das zum Zugriff auf die Redis-Instanz verwendet wird. Dies sollte auf eine zufällige Zeichenkette festgelegt werden, um die Sicherheit zu gewährleisten.
COUCHDB_USER und COUCHDB_PASSWORD	Diese beiden Umgebungsvariablen definieren den CouchDB-Benutzernamen und das Passwort, die zum Zugriff auf den Hauptadministrationsbenutzer verwendet werden. Diese können auch verwendet werden, um auf die Fauxton-Benutzeroberfläche zuzugreifen.

oder als Composer Datei mit ssl

Dazu ein neues Verzeichnis namens certs im Verzeichnis wo die composer Datei liegt / hinsoll erstellen.

```
mkdir certs
```

Nun ins Verzeichnis gehen und das Zertifikat anlegen.

```
cd /home/duffy/budibasedockertemplate/certs
openssl req -newkey rsa:4096 -x509 -sha256 -days 365000 -nodes -out selfsigned.crt -keyout private.key
```

nun die Fragen beantworten.
DE
und der Common Nname als URL.
Bei selbstsigniert eigentlich egal. Hauptsache steht was drin.

Ausgabe:

```
.....++++
.....++++
writing new private key to 'private.key'
-----
You are about to be asked to enter information that will be incorporated
into your certificate request.
What you are about to enter is what is called a Distinguished Name or a DN.
There are quite a few fields but you can leave some blank
For some fields there will be a default value,
If you enter '.', the field will be left blank.
-----
```

```
Country Name (2 letter code) [AU]:DE
State or Province Name (full name) [Some-State]:
Locality Name (eg, city) []:
Organization Name (eg, company) [Internet Widgits Pty Ltd]:
Organizational Unit Name (eg, section) []:
Common Name (e.g. server FQDN or YOUR name) []:budibase.local.lan
Email Address []:
```

Nun wieder in das Verzeichnis wo die composer-datei liegt zurück gehen.

```
cd ..
```

Nun wieder ein Verzeichnis zurück ins Template Verzeichnis und eine neue nginx Config namens "nginx-proxy.conf" anlegen

```
nano nginx-proxy.conf
```

Inhalt dieser Datei:
Client_max_body_size gibt an wie groß eine Datei zum Upload sein darf. Bruacht man größere größen diese anpassen.
Wenn im nginx dieser Fehler im Log auftaucht:

```
Fehler: client intended to send too large bod
```

Nun der Inhalt

```
events {
  worker_connections 1024;
}

http {
  server {
    listen 80;
    return 301 https://$host$request_uri;
  }

  server {
    listen 443 ssl;
    ssl_certificate /etc/ssl/certs/selfsigned.crt;
    ssl_certificate_key /etc/ssl/private/private.key;
```

```
    location / {
      proxy_pass http://budibase:80;
      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;
      client_max_body_size 100M;
    }
  }
}
```

Die composer Datei:

```
version: "3"

services:
  budibase:
    restart: unless-stopped
    image: budibase/budibase:latest
    expose:
      - "80"
    healthcheck:   -> ein heltcheck der 3 mal überprüft ob der dienst läuft wenn nicht wird er als unheatly. markiert. siehe unten bei restart on failure
      test: ["CMD", "curl", "--fail", "http://localhost:80/"]
      interval: 30s
      timeout: 10s
      retries: 3

    volumes:
      - budibase_data:/data   -> Bezieht sich unten auf das Volume, wo die Budibase Daten gespeichert werden sollen, alspo persitentlaufwerk
  nginx:
    image: nginx:stable
    container_name: nginx-proxy
```

```
    volumes:
      - ./nginx-proxy.conf:/etc/nginx/nginx.conf:ro   -> Es können anstatt Verzeichnisse auch Dateien angebene werden. Hoer nginx config. Im lesen Mdous
      - ./certs/selfsigned.crt:/etc/ssl/certs/selfsigned.crt:ro  Hier der Öffentliche Schlüssel für ssl auch lesend
      - ./certs/private.key:/etc/ssl/private/private.key:ro  Hier der Private Schlüssel für ssl auch lesend.
    ports:
      - "80:80" -> port 80 - weitergelietet auf port 80
      - "443:443" -> 443 port auf 443 port
    depends_on:
      - budibase -> braucht budibase gestartet
    restart: unless-stopped   -> wenn oben budibase fehlerhaft ist startet der nginx mit neu

volumes:
  budibase_data:  -> name
    driver: local
    driver_opts:
      type: none
      o: bind
      device: /home/duffy/budibasedata ->  verzeichnis wo die budibase daten liegen auf dem host
```

Und Komplett ohne Erläuterung

```
version: "3"

services:
  budibase:
    restart: unless-stopped
    image: budibase/budibase:latest
    expose:
      - "80"
    healthcheck:
    test: ["CMD", "curl", "--fail", "http://localhost:80/"]
      interval: 30s
      timeout: 10s
      retries: 3
    volumes:
      - budibase_data:/data
  nginx:
    image: nginx:stable
    container_name: nginx-proxy
```

```
    volumes:
      - ./nginx-proxy.conf:/etc/nginx/nginx.conf:ro
      - ./certs/selfsigned.crt:/etc/ssl/certs/selfsigned.crt:ro
      - ./certs/private.key:/etc/ssl/private/private.key:ro
    ports:
      - "80:80"
      - "443:443"
    depends_on:
      - budibase
    restart: unless-stopped

volumes:
  budibase_data:
    driver: local
    driver_opts:
      type: none
      o: bind
      device: /home/duffy/budibasedata
```

Nun den container starten/aktualisieren.

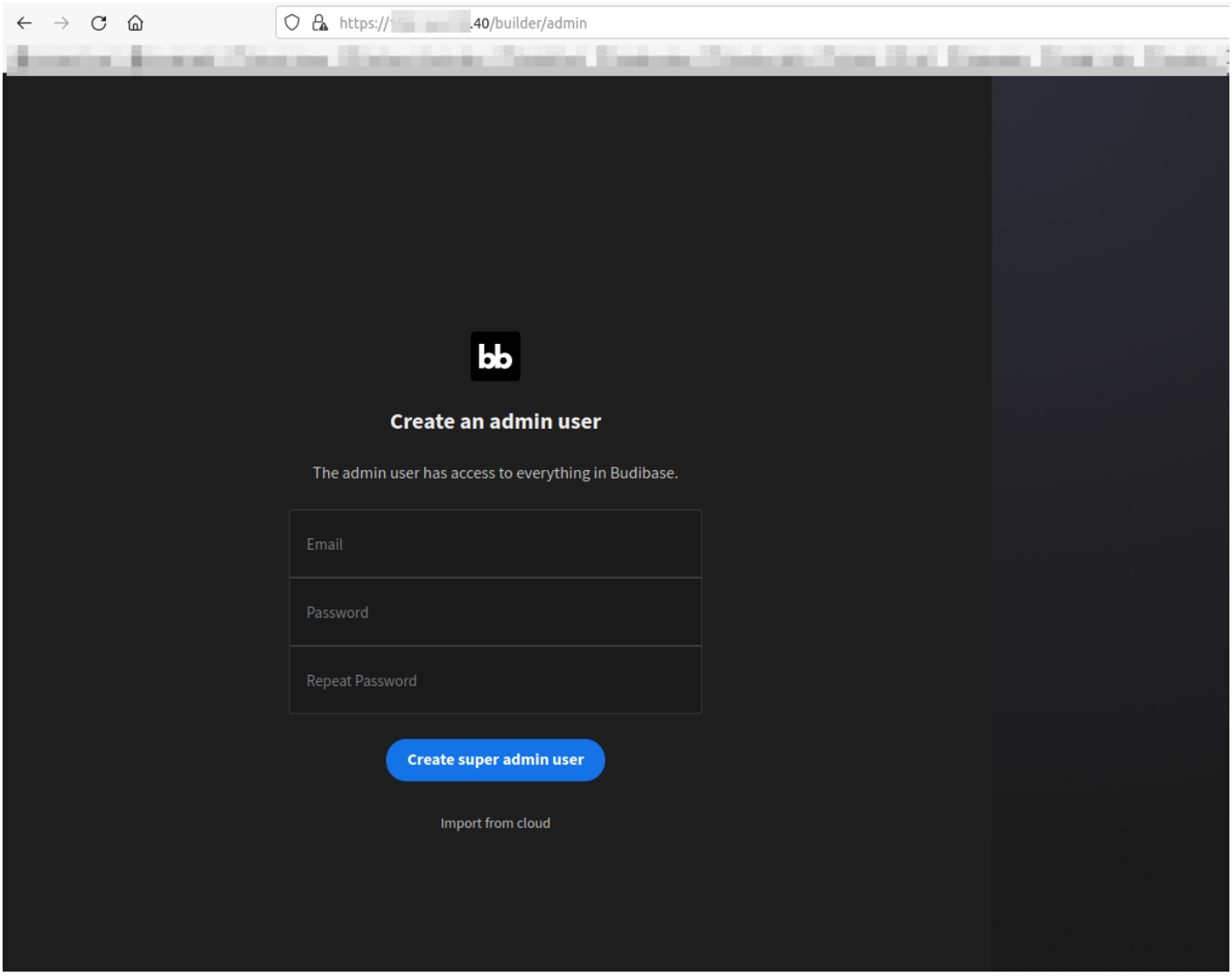
```
docker-compose down
docker-compose up -d
```

Fürs aktualisieren erst down nehmen und dann up.
Für den ersten start nur up.

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

Nun kann ein Webbrowser geöffnet werden, mit https. Das zertifikat beim ersten aufruf naturlich zur ausnahme hinzufügen

https://ip_vom_dockerhost



Nun kann ein Admin Benutzer erstellt werden.