mattermost

- chat claude

chat claude

https://claude.ai/chat/b1885736-ac35-42c7-9bb0-205f08c37415

nginx

Oui, c'est exactement ça ! Mais il y a quelques points importants à gérer :

Architecture multi-instances Mattermost

1. **Côté Docker** (tu dis savoir faire, mais je résume) :

Chaque instance doit avoir :

- Son propre `docker-compose.yml` dans un dossier séparé
- Son propre `.env` avec des valeurs uniques :

```
# Instance 1 : ports 8065 + base postgres
5432APP_PORT=8065POSTGRES_DATA_PATH=./volumes/db1/...# Instance 2 : ports 8066 +
base postgres 5433APP_PORT=8066POSTGRES_DATA_PATH=./volumes/db2/...
```

- Ses propres volumes de données
- **Optionnel** : un réseau Docker séparé pour isoler chaque instance

2. **Côté Nginx** :

Pour chaque instance, tu crées un nouveau fichier de conf avec :### 3. **Certificats SSL avec Certbot** :

Pour chaque nouveau domaine :

```
# Obtenir un certificat pour le nouveau domaine
sudo certbot certonly --nginx -d projet-x.aubaygues.fr
```

```
# Ou plusieurs domaines à la fois
sudo certbot certonly --nginx -d projet-x.aubaygues.fr -d chat-dev.aubaygues.fr
```

Certbot va :

- Créer les certificats dans `/etc/letsencrypt/live/projet-x.aubaygues.fr/`
- Gérer le renouvellement automatique (tous les domaines ensemble)

4. **Activation des configurations** :

```
# Créer les symlinks
sudo ln -s /etc/nginx/sites-available/mattermost-instance1.conf /etc/nginx/sites-enabled/
sudo ln -s /etc/nginx/sites-available/mattermost-instance2.conf /etc/nginx/sites-enabled/

# Tester
sudo nginx -t

# Recharger
sudo systemctl reload nginx
```

Points d'attention importants :

?? **Nommage des upstreams**

Chaque `upstream` doit avoir un nom **unique** :

- `backend_instance1`, `backend_instance2`, etc.
- Ou `backend_groupe`, `backend_projetx`, etc.

?? **Ports Docker uniques**

```
# Instance 1
ports:
  - 8065:8065

# Instance 2
```

```
ports:
  - 8066:8065  # Port hôte 8066 → port container 8065

# Instance 3
ports:
  - 8067:8065
```

?? Variables d'environnement `.env`

Chaque instance doit avoir son propre `MM_SERVICESETTINGS_SITEURL` :

```
# Instance 1
MM_SERVICESETTINGS_SITEURL=https://groupe.aubaygues.fr

# Instance 2
MM_SERVICESETTINGS_SITEURL=https://projet-x.aubaygues.fr
```

Organisation recommandée :

```
/opt/mattermost/
├── instance1-groupe/
│   ├── docker-compose.yml
│   ├── .env (DOMAIN=groupe.aubaygues.fr, APP_PORT=8065)
│   └── volumes/
├── instance2-projetx/
│   ├── docker-compose.yml
│   ├── .env (DOMAIN=projet-x.aubaygues.fr, APP_PORT=8066)
│   └── volumes/
└── instance3-dev/
    ├── docker-compose.yml
    ├── .env (DOMAIN=chat-dev.aubaygues.fr, APP_PORT=8067)
    └── volumes/
```

Script pour faciliter la création :

```
#!/bin/bash
# create-mattermost-instance.sh

INSTANCE_NAME=$1
DOMAIN=$2
PORT=$3

if [ -z "$INSTANCE_NAME" ] || [ -z "$DOMAIN" ] || [ -z "$PORT" ]; then
    echo "Usage: $0 <instance_name> <domain> <port>"
    echo "Exemple: $0 projetx projet-x.aubaygues.fr 8066"
    exit 1
fi

# Créer le dossier
mkdir -p /opt/mattermost/$INSTANCE_NAME
cd /opt/mattermost/$INSTANCE_NAME

# Copier les fichiers de base depuis instance1
cp -r /opt/mattermost/instance1-groupe/docker-compose.yml .
cp -r /opt/mattermost/instance1-groupe/.env .

# Adapter le .env
sed -i "s/DOMAIN=.*/DOMAIN=$DOMAIN/" .env
sed -i "s/APP_PORT=.*/APP_PORT=$PORT/" .env
sed -i "s|POSTGRES_DATA_PATH=.*|POSTGRES_DATA_PATH=./volumes/db/var/lib/postgresql/data|"
.env

# Créer la structure de volumes
mkdir -p volumes/db/var/lib/postgresql/data
mkdir -p volumes/app/mattermost/{config,data,logs,plugins,client/plugins,bleve-indexes}

# Permissions
sudo chown -R 2000:2000 volumes/app/mattermost

echo "□ Instance créée dans /opt/mattermost/$INSTANCE_NAME"
echo "Prochaines étapes:"
echo "1. cd /opt/mattermost/$INSTANCE_NAME"
echo "2. docker-compose up -d"
echo "3. sudo certbot certonly --nginx -d $DOMAIN"
```

```
echo "4. Créer /etc/nginx/sites-available/mattermost-$INSTANCE_NAME.conf"
echo "5. sudo ln -s /etc/nginx/sites-available/mattermost-$INSTANCE_NAME.conf
/etc/nginx/sites-enabled/"
echo "6. sudo nginx -t && sudo systemctl reload nginx"
```

Voilà ! En gros : **oui, c'est aussi simple que tu le penses**, mais attention aux noms d'upstreams et aux ports □□

```
# ===== INSTANCE 2 : projet-x.aubaygues.fr =====
# Fichier : /etc/nginx/sites-available/mattermost-instance2.conf

upstream backend_instance2 {
   server localhost:8066;  # Port différent !
   keepalive 64;
}

server {
   listen 80;
   server_name projet-x.aubaygues.fr;
   return 301 https://$host$request_uri;
}

server {
   listen 443 ssl http2;
   server_name projet-x.aubaygues.fr;

   ssl_certificate /etc/letsencrypt/live/projet-x.aubaygues.fr/fullchain.pem;
   ssl_certificate_key /etc/letsencrypt/live/projet-x.aubaygues.fr/privkey.pem;
   include /etc/letsencrypt/options-ssl-nginx.conf;
   ssl_dhparam /etc/letsencrypt/ssl-dhparams.pem;

   location ~ /api/v[0-9]+/(users/)?websocket$ {
      proxy_set_header Upgrade $http_upgrade;
      proxy_set_header Connection "upgrade";
      proxy_set_header Host $http_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_pass http://backend_instance2;
   }

   location / {
      proxy_set_header Host $http_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_http_version 1.1;
        proxy_pass http://backend_instance2;
    }
}
```