MIMM 2.0 – Deployment Checklist (Lite, Detailní verze pro nelinuxáře)

> Copy–paste návod pro jedno VPS (Hetzner Ubuntu 24.04). Každý krok má příkaz a krátké vysvětlení.

Poznámky: - Připoj se přes SSH z Windows (PowerShell nebo PuTTY) / z macOS (Terminal). - `IP` nahraď veřejnou IP serveru, `your-domain.com` svou doménou. - Pokud se po změně SSH portu odpojíš, připoj se znovu s `-p 2222`.

Co sakra potřebuješ mít ready PŘED začátkem

Než začneš cokoliv instalovat, měl bys mít:

1. **VPS běží** – Hetzner/DigitalOcean/cokoliv, Ubuntu 24.04, min. 2 GB RAM, máš root přístup
2. **Doménu** – koupená a DNS A records nastaveny na IP serveru (pro `your-domain.com`, `www`, `api`)
3. **SSH klíče** – vygenerované na svém PC: `ssh-keygen -t ed25519 -C "tvuj@email.com"`
4. **Silná hesla připravená:**
   - Database password (min. 16 znaků, random)
   - Redis password (min. 16 znaků)
   - JWT secret key (min. 32 znaků, použij `openssl rand -base64 32`)
5. **Email** – platný email pro Let's Encrypt notifikace
6. **Git repo s kódem** – buď public, nebo si nastav SSH deploy key
7. **Frontend build** – zkompilovaný Blazor WASM (složka `dist/` nebo `wwwroot/`)
8. **Volitelné API klíče** (pokud používáš):
   - SendGrid API key (pro emaily)
   - Last.fm API key + shared secret
   - Discogs token

Pro hesla použij password manager (Bitwarden, 1Password, KeePass) – nepiš si je na papír nebo do plaintext souboru.

Pokud tohle nemáš, zastav se TEĎKA a připrav si to. Jinak budeš deployment zdržovat a hledat věci v půlce procesu.

Fáze A: Základ serveru (15–20 min)

1) Přihlášení jako root

```
ssh root@IP
```

2) Aktualizace balíčků (stáhne a nainstaluje dostupné updaty)

```
apt update && apt upgrade -y
```

3) Vytvoření uživatele `mimm` a přidání do sudo (aby nemusel používat root)

```
adduser mimm            # nastavte heslo, stačí Enter pro volitelné údaje
usermod -aG sudo mimm   # dá uživateli práva sudo
```

4) SSH hardening (změna portu, zákaz root a hesel)

```
nano /etc/ssh/sshd_config
# změňte nebo přidejte řádky:
# Port 2222
# PermitRootLogin no
```

```
# PasswordAuthentication no
# Uložte: Ctrl+O, Enter, ukončete: Ctrl+X
systemctl restart sshd
```

Co to dělá: port 2222 sníží šum botů, zakáže login jako root a zakáže hesla (jen klíče).

5) Firewall UFW (povolí jen SSH+HTTP+HTTPS, zbytek blokne)

```
ufw allow 2222/tcp
ufw allow 80/tcp
ufw allow 443/tcp
ufw enable   # potvrďte "y"
ufw status verbose
```

6) Fail2Ban (ochrana proti brute force na SSH a Nginx)

```
apt install -y fail2ban
systemctl enable --now fail2ban
fail2ban-client status
```

7) Přihlášení jako nový uživatel (otestuj, že vše funguje)

```
ssh mimm@IP -p 2222
```

Fáze B: Docker + Nginx (15–20 min)

1) Docker + compose plugin (instalační skript Dockeru + plugin Compose)

```
curl -fsSL https://get.docker.com -o get-docker.sh
sh get-docker.sh
apt install -y docker-compose-plugin
```

2) Přidání uživatele do docker group (aby mohl spouštět docker bez sudo)

```
sudo usermod -aG docker mimm
# pak se odhlásit a přihlásit, jinak se skupina neprojeví
```

3) Test Dockeru

```
docker run hello-world
```

4) Nginx instalace a vypnutí default site

```
sudo apt install -y nginx
sudo rm /etc/nginx/sites-enabled/default
sudo nginx -t   # test konfigurace (zatím prázdná, ale ok)
```

Fáze C: Certy (Let's Encrypt, 10 min)

1) Certbot instalace

```
sudo apt install -y certbot python3-certbot-nginx
```

2) Získání certů (NGINX musí běžet)

```
sudo certbot certonly --nginx \
  -d your-domain.com \
  -d www.your-domain.com \
  -d api.your-domain.com
```

Co to dělá: vystaví HTTPS certifikáty pro tři hostname.

3) Ověření obnovy

```
sudo certbot renew --dry-run
```

Fáze D: Aplikace (20–30 min)

1) Repo stáhnout / nahrát

```
cd /home/mimm
git clone <repo-url> mimm-app    # nebo nahrát SFTP do /home/mimm/mimm-app
cd mimm-app
```

2) `.env` vytvořit a zamknout (jen na serveru)

```
nano .env
# vlož hodnoty (viz LITE plán: DB, JWT, URL, atd.)
chmod 600 .env
```

Ukázka obsahu `.env` (nahraď vlastními hodnotami):

```
POSTGRES_USER=mimmuser
POSTGRES_PASSWORD=STRONG_DB_PASS
POSTGRES_DB=mimm

JWT_SECRET_KEY=AT_LEAST_32_CHARS_SECRET
JWT_ISSUER=https://api.your-domain.com
JWT_AUDIENCE=mimm-frontend

FRONTEND_URL=https://your-domain.com

# Pokud používáš Redis, jinak smaž:
REDIS_PASSWORD=STRONG_REDIS_PASS

# Volitelné integace (nech prázdné, pokud nepoužiješ)
SENDGRID_API_KEY=
SENDGRID_FROM_EMAIL=noreply@your-domain.com
```

3) Docker Compose (připrav docker-compose.prod.yml; Redis dej jen pokud ho používáš)

- Pokud máš hotový soubor z repa, nic nedělej.
- Jinak použij šablonu z `DEPLOYMENT_PLAN_LITE.md`.

4) Nginx configy

```
# Backend config ulož do /etc/nginx/sites-available/mimm-backend
# Frontend config ulož do /etc/nginx/sites-available/mimm-frontend
sudo ln -s /etc/nginx/sites-available/mimm-backend /etc/nginx/sites-enabled/
sudo ln -s /etc/nginx/sites-available/mimm-frontend /etc/nginx/sites-enabled/
sudo nginx -t
sudo nginx -s reload
```

Minimal backend config (`/etc/nginx/sites-available/mimm-backend`):

```
server {
  listen 80;
  server_name api.your-domain.com;
  location /.well-known/acme-challenge/ { root /var/www/certbot; }
```

```
  return 301 https://$host$request_uri;
}

server {
  listen 443 ssl http2;
  server_name api.your-domain.com;

  ssl_certificate /etc/letsencrypt/live/api.your-domain.com/fullchain.pem;
  ssl_certificate_key /etc/letsencrypt/live/api.your-domain.com/privkey.pem;

  add_header Strict-Transport-Security "max-age=63072000; includeSubDomains; preload" always;
  add_header X-Frame-Options "SAMEORIGIN" always;
  add_header X-Content-Type-Options "nosniff" always;
  server_tokens off;

  client_max_body_size 10M;
  upstream mimm_backend { server 127.0.0.1:5001; keepalive 16; }

  location / {
    proxy_pass http://mimm_backend;
    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;
  }

  location /hubs/ {
    proxy_pass http://mimm_backend;
    proxy_http_version 1.1;
    proxy_set_header Upgrade $http_upgrade;
    proxy_set_header Connection "upgrade";
    proxy_set_header Host $host;
    proxy_set_header X-Forwarded-For $proxy_add_x_forwarded_for;
    proxy_set_header X-Forwarded-Proto $scheme;
  }

  location /health {
    proxy_pass http://mimm_backend;
    access_log off;
  }
}
```

Minimal frontend config (`/etc/nginx/sites-available/mimm-frontend`):

```
server {
  listen 80;
  server_name your-domain.com www.your-domain.com;
  location /.well-known/acme-challenge/ { root /var/www/certbot; }
  return 301 https://your-domain.com$request_uri;
}

server {
  listen 443 ssl http2;
  server_name your-domain.com;
```

```
  ssl_certificate /etc/letsencrypt/live/your-domain.com/fullchain.pem;
  ssl_certificate_key /etc/letsencrypt/live/your-domain.com/privkey.pem;

  add_header Strict-Transport-Security "max-age=63072000; includeSubDomains; preload" always;
  add_header X-Frame-Options "SAMEORIGIN" always;
  add_header X-Content-Type-Options "nosniff" always;
  server_tokens off;

  root /var/www/mimm-frontend;
  index index.html;

  location / {
    try_files $uri $uri/ /index.html =404;
  }
}
```

5) Build & run

```
docker compose -f docker-compose.prod.yml up -d
```

6) Migrace databáze

```
docker compose -f docker-compose.prod.yml run --rm backend \
  dotnet ef database update --no-build
```

Fáze E: Smoke test (10 min)

1) Backend health

```
curl -I https://api.your-domain.com/health    # očekávej HTTP/1.1 200 OK
```

2) Frontend

- Otevři https://your-domain.com v prohlížeči, zkontroluj že se načte bez chyb.

3) Login/registrace

- Založ testovací účet, přihlášení musí projít.

Fáze F: Backup (5 min)

1) Složka pro backupy

```
mkdir -p ~/backups
```

2) Jednorázový dump (spusť kdykoli)

```
docker exec mimm-postgres pg_dump -U mimmuser mimm | \
  gzip > ~/backups/mimm_db_$(date +%Y%m%d).sql.gz
```

3) Denní cron v 2:00

```
crontab -e
# přidej řádek (pozor na backslashy):
0 2 * * * docker exec mimm-postgres pg_dump -U mimmuser mimm | gzip > ~/backups/mimm_db_\%Y\%m
```

Fáze G: Minimum monitoringu (5 min)

1) UptimeRobot

- Přidej HTTP(S) check na `https://api.your-domain.com/health`.

2) Rychlé logy

```
docker compose -f docker-compose.prod.yml logs --tail=200
```

Volitelné (až bude čas)

- Redis přidej, jen pokud appka potřebuje cache/SignalR scale-out.
- Docker Bench / Lynis (security audit) až později.
- Off-site backup sync (rsync/S3) až později.
- Rychlý výkonový test: `ab -n 100 -c 5 https://api.your-domain.com/health`.

Go/No-Go

☐ Vše zelené → GO
☐ Něco chybí → NO-GO, doplnit