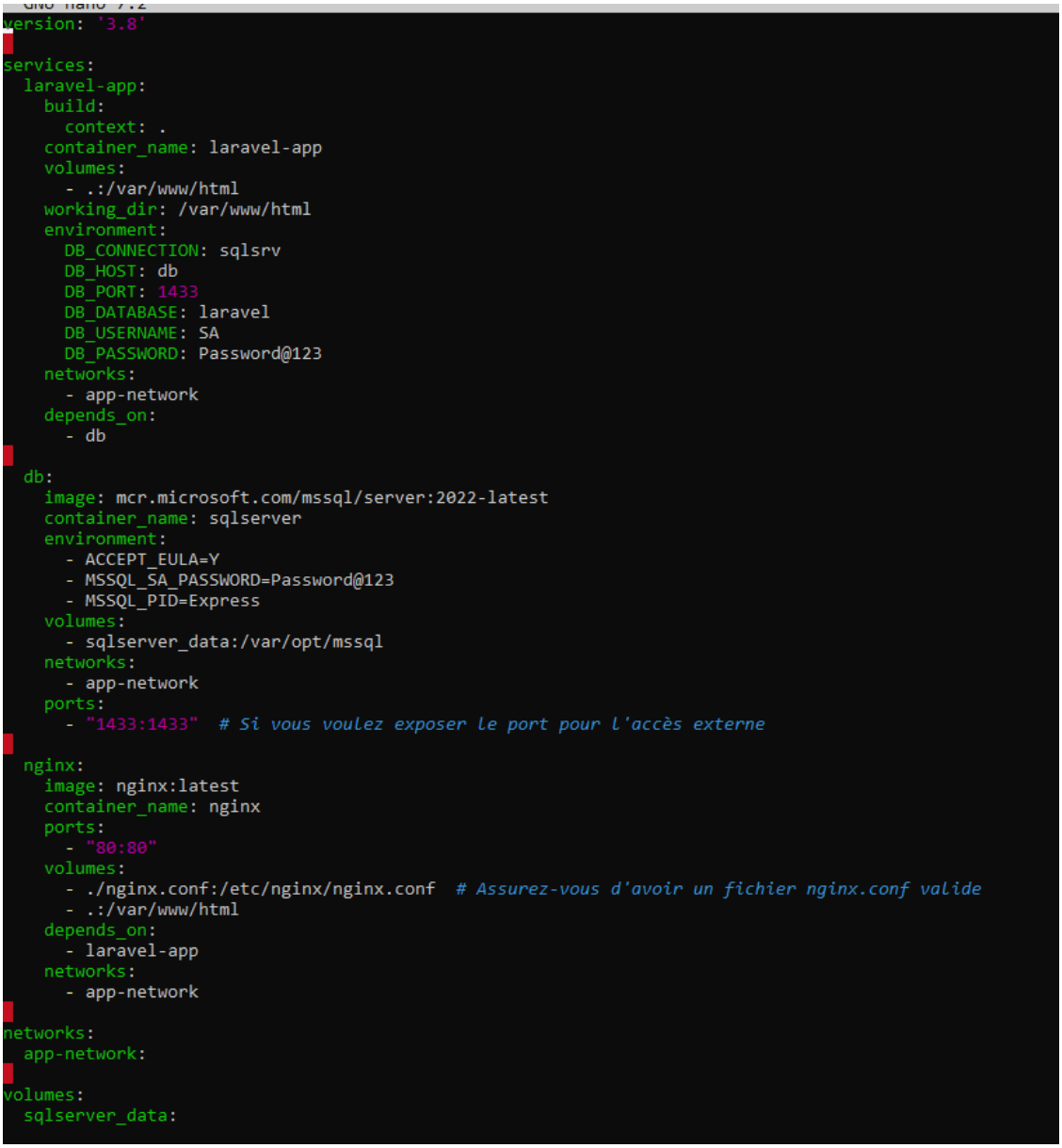
Docker-compose.yml :

```
version: '3.8'

services:
 laravel-app:
  build:
   context: .
  container_name: laravel-app
  volumes:
   - .:/var/www/html
  working_dir: /var/www/html
  environment:
   DB_CONNECTION: sqlsrv
   DB_HOST: db
   DB_PORT: 1433
   DB_DATABASE: laravel
   DB_USERNAME: SA
   DB_PASSWORD: Password@123
  networks:
   - app-network
  depends_on:
   - db

 db:
  image: mcr.microsoft.com/mssql/server:2022-latest
  container_name: sqlserver
  environment:
   - ACCEPT_EULA=Y
   - MSSQL_SA_PASSWORD=Password@123
   - MSSQL_PID=Express
  volumes:
```

```
    - sqlserver_data:/var/opt/mssql
  networks:
    - app-network
  ports:
    - "1433:1433"  # Si vous voulez exposer le port pour l'accès externe

 nginx:
  image: nginx:latest
  container_name: nginx
  ports:
    - "80:80"
  volumes:
    - ./nginx.conf:/etc/nginx/nginx.conf  # Assurez-vous d'avoir un fichier nginx.conf valide
    - .:/var/www/html
  depends_on:
    - laravel-app
  networks:
    - app-network

networks:
 app-network:

volumes:
 sqlserver_data:
```



```yaml
version: '3.8'

services:
  laravel-app:
    build:
      context: .
    container_name: laravel-app
    volumes:
      - .:/var/www/html
    working_dir: /var/www/html
    environment:
      DB_CONNECTION: sqlsrv
      DB_HOST: db
      DB_PORT: 1433
      DB_DATABASE: laravel
      DB_USERNAME: SA
      DB_PASSWORD: Password@123
    networks:
      - app-network
    depends_on:
      - db

  db:
    image: mcr.microsoft.com/mssql/server:2022-latest
    container_name: sqlserver
    environment:
      - ACCEPT_EULA=Y
      - MSSQL_SA_PASSWORD=Password@123
      - MSSQL_PID=Express
    volumes:
      - sqlserver_data:/var/opt/mssql
    networks:
      - app-network
    ports:
      - "1433:1433"  # Si vous voulez exposer le port pour l'accès externe

  nginx:
    image: nginx:latest
    container_name: nginx
    ports:
      - "80:80"
    volumes:
      - ./nginx.conf:/etc/nginx/nginx.conf  # Assurez-vous d'avoir un fichier nginx.conf valide
      - .:/var/www/html
    depends_on:
      - laravel-app
    networks:
      - app-network

networks:
  app-network:

volumes:
  sqlserver_data:
```

Dockerfile:

FROM php:8.2-fpm

Installer les dépendances nécessaires

RUN apt-get update && apt-get install -y \

curl \

gnupg \

ca-certificates \

unixodbc-dev \

gcc \

```
    g++ \
    make \
    autoconf \
    libc-dev \
    pkg-config \
    libxml2-dev \
    bash \
    libcurl4-openssl-dev \
    libssl-dev

# Ajouter la clé GPG Microsoft
RUN curl -fsSL https://packages.microsoft.com/keys/microsoft.asc | gpg --dearmor -o /usr/share/keyrings/microsoft-prod.gpg

# Ajouter le dépôt Microsoft pour Debian 12 (bookworm)
RUN curl https://packages.microsoft.com/config/debian/12/prod.list | tee /etc/apt/sources.list.d/mssql-release.list

# Mettre à jour les dépôts
RUN apt-get update

# Installer le ODBC driver pour SQL Server
RUN ACCEPT_EULA=Y apt-get install -y msodbcsql18

# Installer les outils mssql (optionnel)
RUN ACCEPT_EULA=Y apt-get install -y mssql-tools18

# Ajouter le chemin mssql-tools au PATH (optionnel)
RUN echo 'export PATH="$PATH:/opt/mssql-tools18/bin"' >> ~/.bashrc

# Installer les extensions SQLSRV et PDO_SQLSRV
RUN pecl install sqlsrv pdo_sqlsrv \
```

&& docker-php-ext-enable sqlsrv pdo_sqlsrv

Nettoyer les fichiers inutiles pour alléger l'image

RUN apt-get clean && rm -rf /var/lib/apt/lists/*

Assurez-vous que les bonnes permissions et la configuration sont en place pour le répertoire de l'application

WORKDIR /var/www/html

```
FROM php:8.2-fpm

# Installer les dépendances nécessaires
RUN apt-get update && apt-get install -y \
    curl \
    gnupg \
    ca-certificates \
    unixodbc-dev \
    gcc \
    g++ \
    make \
    autoconf \
    libc-dev \
    pkg-config \
    libxml2-dev \
    bash \
    libcurl4-openssl-dev \
    libssl-dev

# Ajouter la clé GPG Microsoft
RUN curl -fsSL https://packages.microsoft.com/keys/microsoft.asc | gpg --dearmor -o /usr/share/keyrings/microsoft-prod.gpg

# Ajouter le dépôt Microsoft pour Debian 12 (bookworm)
RUN curl https://packages.microsoft.com/config/debian/12/prod.list | tee /etc/apt/sources.list.d/mssql-release.list

# Mettre à jour les dépôts
RUN apt-get update

# Installer le ODBC driver pour SQL Server
RUN ACCEPT_EULA=Y apt-get install -y msodbcsql18

# Installer les outils mssql (optionnel)
RUN ACCEPT_EULA=Y apt-get install -y mssql-tools18

# Ajouter le chemin mssql-tools au PATH (optionnel)
RUN echo 'export PATH="$PATH:/opt/mssql-tools18/bin"' >> ~/.bashrc

# Installer les extensions SQLSRV et PDO_SQLSRV
RUN pecl install sqlsrv pdo_sqlsrv \
    && docker-php-ext-enable sqlsrv pdo_sqlsrv

# Nettoyer les fichiers inutiles pour alléger l'image
RUN apt-get clean && rm -rf /var/lib/apt/lists/*

# Assurez-vous que les bonnes permissions et la configuration sont en place pour le répertoire de l'application
WORKDIR /var/www/html
```

Fichier .env :

```
DB_CONNECTION=sqlsrv
DB_HOST=db
DB_PORT=1433
DB_DATABASE=laravel
DB_USERNAME=SA
DB_PASSWORD=Password@123

SESSION_DRIVER=database
File Name to Write: .env
```

NGINX :

```
user nginx;
worker_processes auto;
pid /run/nginx.pid;

events {
  worker_connections 1024;
}

http {
  include /etc/nginx/mime.types;
  default_type application/octet-stream;

  sendfile on;
  tcp_nopush on;
  tcp_nodelay on;
  keepalive_timeout 65;
  types_hash_max_size 2048;

  # Configuration du serveur pour Laravel
  server {
    listen 80;
    server_name localhost;
    root /var/www/html/public;

    index index.php index.html index.htm;

    location / {
      try_files $uri $uri/ /index.php?$query_string;
    }
```

```
    location ~ \.php$ {
        include fastcgi_params;
        fastcgi_pass laravel-app:9000;
        fastcgi_index index.php;
        fastcgi_param SCRIPT_FILENAME $document_root$fastcgi_script_name;
    }

    # Bloquer l'accès aux fichiers sensibles
    location ~ /\.(env|htaccess|git|svn|dockerignore|env.example) {
        deny all;
    }
  }
}
```

docker compose up -d –build

Mise au point:

Autoriser confiance certificate :

config/database.php

```
        'sqlsrv' => [
            'driver' => 'sqlsrv',
            'url' => env('DB_URL'),
            'host' => env('DB_HOST', 'localhost'),
            'port' => env('DB_PORT', '1433'),
            'database' => env('DB_DATABASE', 'laravel'),
            'username' => env('DB_USERNAME', 'root'),
            'password' => env('DB_PASSWORD', ''),
            'charset' => env('DB_CHARSET', 'utf8'),
            'prefix' => '',
            'prefix_indexes' => true,
            // 'encrypt' => env('DB_ENCRYPT', 'yes'),
             'trust_server_certificate' => env('DB_TRUST_SERVER_CERTIFICATE', 'true'),
        ],
```

Droit au dossier storage :

sudo chmod -R 775 storage

sudo chown -R www-data:www-data storage

Une fois le docker-compose et file fonctionnel faire migrer base de donnée

docker-compose exec laravel-app php artisan migrate

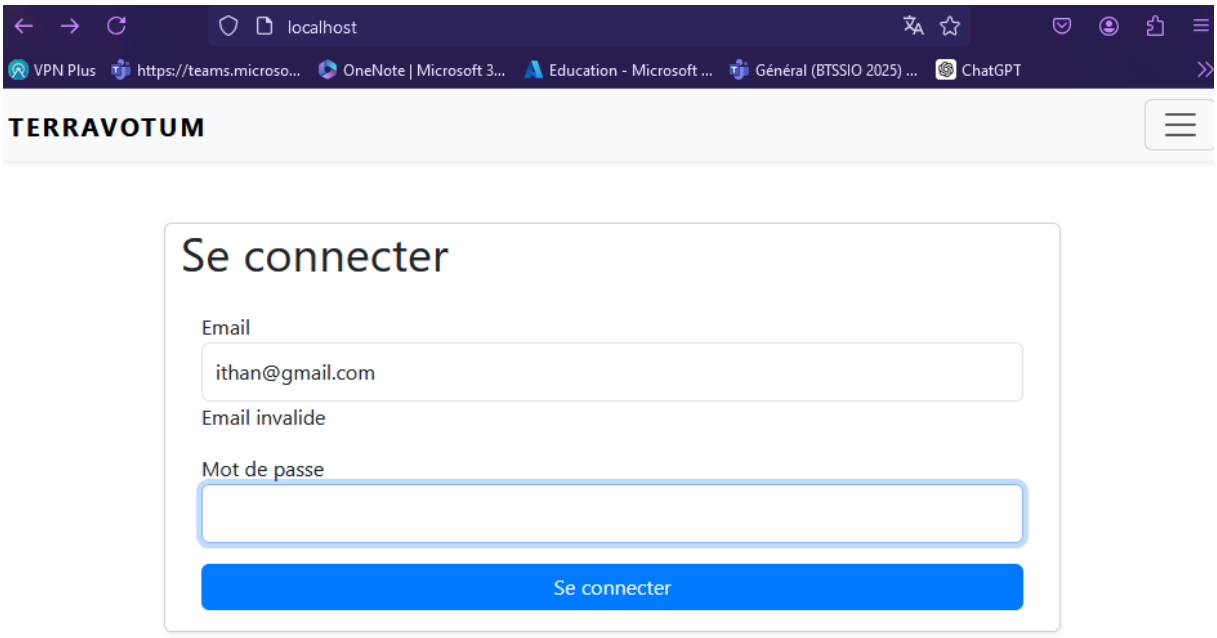
```
urtic@ASUS-ZENBOOK:~/laravel$ docker-compose exec laravel-app php artisan migrate

  INFO  Preparing database.

Creating migration table ..... 18.49ms DONE

  INFO  Running migrations.

0001_01_01_000000_create_users_table ..... 22.44ms DONE
0001_01_01_000001_create_cache_table ..... 12.41ms DONE
0001_01_01_000002_create_jobs_table ..... 16.61ms DONE
2024_10_16_082234_mouvement ..... 6.22ms DONE
2024_10_16_082339_parti ..... 68.76ms DONE
2024_10_16_082459_candidat ..... 4.35ms DONE
2024_10_17_082119_member ..... 48.12ms DONE
2024_10_17_170000_region ..... 4.55ms DONE
2024_10_17_172227_departement ..... 5.73ms DONE
```

docker-compose exec laravel-app php artisan db:seed --class=DatabaseSeeder

```
urtic@ASUS-ZENBOOK:~/laravel$ docker-compose exec laravel-app php artisan db:seed --class=DatabaseSeeder

  INFO  Seeding database.

Database\Seeders\RegionSeeder ..... RUNNING
Database\Seeders\RegionSeeder ..... 29 ms DONE

Database\Seeders\MouvementSeeder ..... RUNNING
Database\Seeders\MouvementSeeder ..... 30 ms DONE

Database\Seeders\PartiSeeder ..... RUNNING
Database\Seeders\PartiSeeder ..... 64 ms DONE
```

docker-compose exec laravel-app php artisan serve

```
urtic@ASUS-ZENBOOK:~/laravel$ docker-compose exec laravel-app php artisan serve

  INFO  Server running on [http://127.0.0.1:8000].

  Press Ctrl+C to stop the server
```



REMARQUE :

SI on veut refaire un docker compose ne pas oublier de supprimer le volume car sinon il vas reutiliser les anciennes données