本番PC 簡易セットアップ手順

本番PC（10.0.1.232）でkaizen_ppアプリを起動する手順

データベースとメディアファイルの移行は既に完了している前提です。

前提条件

☑ 既に完了している内容:

- kaizen_dbデータベース作成済み
- データベースにデータ移行済み
- 既存Docker MySQL稼働中
- 既存Adminer稼働中（ポート8082）

セットアップ手順

1. プロジェクトファイルの配置

開発PCから本番PCにプロジェクトファイルを転送するか、Gitでクローン:

方法A: Gitでクローン（推奨）

```
cd C:\
git clone https://github.com/oususen/kaizen_pp.git
cd kaizen_pp
```

方法B: ファイル転送

- 開発PCの `kaizen_pp` フォルダを本番PCの `C:\kaizen_pp` にコピー

---

2. 環境変数ファイル（.env）の作成

```
cd C:\kaizen_pp
copy .env.example .env
notepad .env
```

`.env` ファイルの内容:

```
# 既存MySQLコンテナに接続
PRIMARY_DB_HOST=mysql
PRIMARY_DB_PORT=3306
PRIMARY_DB_USER=root
```

```
PRIMARY_DB_PASSWORD=既存MySQLのrootパスワード
PRIMARY_DB_NAME=kaizen_db
```

重要: `PRIMARY_DB_PASSWORD` に既存Docker MySQLのrootパスワードを入力してください。

---

3. メディアファイルの配置

開発PCから `media` フォルダを転送:

```
# 開発PCでmediaフォルダを圧縮
# 本番PCに転送して展開

# または直接コピー
# 開発PC: d:\kaizen_pp\media
# ↓
# 本番PC: C:\kaizen_pp\media
```

確認:

```
powershell -Command "Test-Path C:\kaizen_pp\media"
# True が返ればOK
```

---

4. 既存MySQLネットワークに接続（重要）

既存のMySQLコンテナが使用しているDockerネットワーク名を確認:

```
docker network ls
```

既存MySQLコンテナのネットワークを確認:

```
docker inspect <既存MySQLコンテナ名> | findstr NetworkMode
```

docker-compose.prod.yml の修正

ファイルの最後にある networks セクションのコメントを外して編集:

```
networks:
  existing_network:
    external: true
    name: 既存のネットワーク名  # ← 上で確認したネットワーク名を入力
```

そして、backendサービスに networks を追加:

```
services:
  backend:
    # ... 既存の設定 ...
    networks:
      - existing_network  # ← 追加

  frontend:
    # ... 既存の設定 ...
    networks:
      - existing_network  # ← 追加
```

---

5. Dockerコンテナの起動

```
cd C:\kaizen_pp

# コンテナをビルド＆起動
docker-compose -f docker-compose.prod.yml up -d --build
```

初回ビルドは5〜10分かかる場合があります。

---

6. 起動確認

```
# コンテナの状態確認
docker-compose -f docker-compose.prod.yml ps
```

以下の2つのコンテナが `Up` になっていればOK:

- `kaizen_backend_prod`
- `kaizen_frontend_prod`

ログの確認:

```
docker-compose -f docker-compose.prod.yml logs -f
```

エラーがないか確認。`Ctrl + C` で終了。

---

7. 動作確認

ブラウザでアクセス:

本番PC上:

- フロントエンド: http://localhost:8503
- バックエンドAPI: http://localhost:8083/api/

開発PCから:

- フロントエンド: http://10.0.1.232:8503
- バックエンドAPI: http://10.0.1.232:8083/api/

確認項目:

- ☑ ログイン画面が表示される
- ☑ ログインできる
- ☑ 提案一覧が表示される
- ☑ 画像が表示される

トラブルシューティング

エラー: データベースに接続できない

原因: `.env` のパスワードが間違っているか、ネットワーク設定が不適切

解決方法:

1. `.env` の `PRIMARY_DB_PASSWORD` を確認
2. Adminer（http://localhost:8082）で接続確認
3. docker-compose.prod.yml のネットワーク設定を確認

```
# 既存MySQLコンテナのネットワーク確認
docker inspect <既存MySQLコンテナ名> --format='{{json .NetworkSettings.Networks}}'
```

エラー: 画像が表示されない

原因: `media` フォルダが正しく配置されていない

解決方法:

```
# mediaフォルダの確認
powershell -Command "Test-Path C:\kaizen_pp\media"
powershell -Command "(Get-ChildItem -Path C:\kaizen_pp\media -Recurse -File |
Measure-Object).Count"

# コンテナ再起動
docker-compose -f docker-compose.prod.yml restart backend
```

コンテナが起動しない

```
# ログでエラー確認
docker-compose -f docker-compose.prod.yml logs backend
docker-compose -f docker-compose.prod.yml logs frontend

# コンテナを削除して再ビルド
docker-compose -f docker-compose.prod.yml down
docker-compose -f docker-compose.prod.yml up -d --build
```

よく使うコマンド

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

# 停止
docker-compose -f docker-compose.prod.yml down

# 再起動
docker-compose -f docker-compose.prod.yml restart

# ログ確認
docker-compose -f docker-compose.prod.yml logs -f

# 状態確認
docker-compose -f docker-compose.prod.yml ps

# バックエンドのみ再起動
docker-compose -f docker-compose.prod.yml restart backend
```

データベース接続情報（確認用）

Adminer（http://10.0.1.232:8082）でkaizen_dbに接続:

- サーバー: `mysql`
- ユーザー名: `root`
- パスワード: 既存MySQLのrootパスワード
- データベース: `kaizen_db`

完了！

これで本番PCでkaizen_ppアプリが稼働します。

アクセスURL:

- フロントエンド: http://10.0.1.232:8503
- バックエンドAPI: http://10.0.1.232:8083/api/

- Adminer: http://10.0.1.232:8082