Hướng dẫn triển khai ASP.NET Core API lên Azure

Cách 1: Sử dụng GitHub Actions để triển khai lên Azure App Service

1. **Tạo một ứng dụng Azure App Service:**
   - Đăng nhập vào Azure Portal.
   - Tạo một Azure App Service mới bằng cách chọn `Create a resource > Web > App Service`.
   - Điền thông tin cơ bản như tên, vùng địa lý, và lựa chọn nền tảng là `.NET Core`.
2. **Cấu hình GitHub Actions:**
   - Trên repository GitHub của bạn, tạo một workflow mới hoặc sửa đổi workflow hiện tại trong thư mục `.github/workflows`.

```yaml
name: Build and deploy .NET Core application to Web App TourishApi20240305102130
on:
push:
    branches:
    - master
env:
AZURE_WEBAPP_NAME: TourishApi20240305102130
AZURE_WEBAPP_PACKAGE_PATH: WebApplication1/publish
CONFIGURATION: Release
DOTNET_CORE_VERSION: 7.0.x
WORKING_DIRECTORY: WebApplication1
AZURE_DATABASE_STRING: ${{ secrets.AZURE_DATABASE_STRING }}
AZURE_REDIS_STRING: ${{ secrets.AZURE_REDIS_STRING }}
AZURE_BLOB_STRING: ${{ secrets.AZURE_BLOB_STRING }}

jobs:
build:
    runs-on: ubuntu-latest
    steps:
    - uses: actions/checkout@v4
    - name: Setup .NET SDK
    uses: actions/setup-dotnet@v3
    with:
```

```
            dotnet-version: ${{ env.DOTNET_CORE_VERSION }}
        - name: Restore
        run: dotnet restore "${{ env.WORKING_DIRECTORY }}"
        - name: Build
        run: dotnet build "${{ env.WORKING_DIRECTORY }}" --configuration ${{ env.CONFIGURATI
        - name: Test
        run: dotnet test "${{ env.WORKING_DIRECTORY }}" --no-build
        - name: Publish
        run: dotnet publish "${{ env.WORKING_DIRECTORY }}" --configuration ${{ env.CONFIGURA
        - name: Publish Artifacts
        uses: actions/upload-artifact@v3
        with:
            name: webapp
            path: ${{ env.AZURE_WEBAPP_PACKAGE_PATH }}
    deploy:
        runs-on: ubuntu-latest
        needs: build
        steps:
        - name: Download artifact from build job
        uses: actions/download-artifact@v3
        with:
            name: webapp
            path: ${{ env.AZURE_WEBAPP_PACKAGE_PATH }}
        - name: Deploy to Azure WebApp
        uses: azure/webapps-deploy@v2
        with:
            app-name: ${{ env.AZURE_WEBAPP_NAME }}
            publish-profile: ${{ secrets.TourishApi20240305102130_4D47 }}
            package: ${{ env.AZURE_WEBAPP_PACKAGE_PATH }}
        env:
            AZURE_DATABASE_STRING: ${{ secrets.AZURE_DATABASE_STRING }}
            AZURE_REDIS_STRING: ${{ secrets.AZURE_REDIS_STRING }}
            AZURE_BLOB_STRING: ${{ secrets.AZURE_BLOB_STRING }}
```

- Đảm bảo rằng bạn đã thiết lập `AZURE_WEBAPP_PUBLISH_PROFILE` trong cài đặt `Secrets` của repository để có quyền truy cập vào Azure.

3. **Triển khai:**
   - Mỗi khi có sự thay đổi trên nhánh `main` (hoặc nhánh được thiết lập trong `on`), GitHub Actions sẽ tự động build và triển khai API của bạn lên Azure App Service.

Cách 2: Tạo Docker image và triển khai local

1. **Build và tạo Docker image:**
   - Mở terminal và di chuyển vào thư mục dự án API của bạn.
   - Xây dựng project .NET Core API:

```
dotnet build --configuration Release
```

   - Tạo Docker image từ project .NET Core API:

```
docker build -t your-image-name .
```

2. **Triển khai bằng Docker:**
    - Chạy Docker container từ image vừa tạo:

```
docker run -d -p 8080:80 your-image-name
```

    - Điều chỉnh cổng (`-p`) nếu cần thiết.

3. **Hoàn tất:**
    - Sau khi triển khai, API sẽ được chạy trên cổng được chỉ định. Truy cập vào `http://localhost:8080` (hoặc cổng bạn đã chỉ định) để kiểm tra API đã được triển khai thành công.

Thông qua các hướng dẫn này, bạn có thể triển khai ASP.NET Core API của mình lên Azure App Service sử dụng GitHub Actions hoặc triển khai local bằng Docker, tùy thuộc vào yêu cầu và môi trường của dự án.

Mô tả các biến môi trường:

```
• AZURE_BLOB_STRING: Connection string để kết nối tới blob storage, dùng
để lưu trữ file (Ảnh, bài viết giới thiệu,...).

• AZURE_DATABASE_STRING: Connection string để kết nối tới Database.
Với Database mới, có thể dùng bộ công cụ CLI của Entity framework để tạo
Database.

• AZURE_REDIS_STRING: Connection string để kết nối tới Redis, dùng để
cache các kết quả tìm kiếm và hiển thị cho người dùng.

• MAIL_SMTP_EMAIL: Tài khoản mail SMTP, dùng để thiết lập tài khoản để
gửi mail thông báo tới các người dùng.

• MAIL_SMTP_PASSWORD: Mật khẩu mail SMTP, dùng để thiết lập tài khoản
để gửi mail thông báo tới các người dùng.

• PAY_OS_API_KEY, PAY_OS_CHECKSUM_KEY, PAY_OS_CLIENT_ID:
Cấu hình thiết lập thanh toán tour tới PayOS, để tìm hiểu cách tích hợp PayOS
vui lòng truy cập: https://PayOS.vn/ để biết thêm thông tin.

• PAY_OS_SERVICE_API_KEY, PAY_OS_SERVICE_CHECKSUM_KEY,
PAY_OS_SERVICE_CLIENT_ID: Cấu hình thiết lập thanh toán dịch vụ tới
PayOS.

• FACEBOOK_CLIENT_ID: Client id hỗ trợ xác thực token đăng nhập từ Face-
book.

• FACEBOOK_SECRET_KEY: Secret key hỗ trợ xác thực token đăng nhập từ
Facebook.


• GOOGLE_CHAT_KEY: Chat key để tạo chat bot đơn giản, được hỗ trợ bởi
```

```
Gemini AI APIs cung cấp từ Google.

• MAIL_SMTP_PASSWORD: Mật khẩu địa chỉ mail chính của hệ thống, dùng
để gửi mail xác nhận tài khoản, khôi phục tài khoản, hóa đơn sau thanh toán
tới địa chỉ mail của người dùng.
```