

Flask Server Üzerinden İstek Alan ve Docker Konteynerda Çalışan RASA Bot

Bu raporun içeriği şu şekildedir; şu ana kadar geliştirdiğim RASA botunu bir Dockerfile ile Dockerize edip bir konteyner üzerinde çalıştırmaya çalıştım. Ardından flask ile yerel bir sunucu yazdım ve yerel bir port üzerinden API olarak botu çalıştıran bu sayede flask server üzerinden iletişime geçebilen bir sistem tasarladım. Rapor bu adımları nasıl gerçekleştirdiğimi, takıldığım noktaları ve bazı önemli noktaları içerir.

Öncelikle daha anlaşılır olması açısından projemin dosya yapısını şu şekilde buraya koyayım ki bazı noktalarda neyi nerede yaptığımızı anlayabilelim.

Project tree:

```
frstRasa/
├── actions
├── app
├── config.yml
├── credentials.yml
├── data
├── docker-compose.yml
├── Dockerfile
├── domain.yml
├── endpoints.yml
├── models
├── requirements.txt
├── Server
│   └── flaskServer.py
└── tests
```

frstRasa yazdığım ilk botun bulunduğu dizinin adıdır. `docker-compose.yml`, `Dockerfile`, `requirements.txt` ve diziniyle beraber (Server) `flaskServer.py` dosyaları sonradan eklendi. Onun dışındaki dosya ve dizinler zaten her RASA projesinde bulunan dizinlerdir.

actions: Rasa botu için özel eylemleri (actions) tanımlayan dosyaların bulunduğu dizin.

app: Docker konteynerında botun kodlarının kopyalandığı dizin. (Dockerfile'da belirtilen COPY . /app komutu ile bu dizine kopyalanıyor.)

config.yml: Rasa'nın yapılandırma ayarlarını içeren dosya.

data: Eğitim verilerini içeren dizin (NLU ve dialog verileri).

docker-compose.yml: Docker Compose yapılandırma dosyası.

Dockerfile: Docker imajını oluşturmak için kullanılan yapılandırma dosyası.

domain.yml: Rasa botunun domain ayarlarını (intents, entities, actions vb.) içeren dosya.

endpoints.yml: Rasa'nın HTTP sunucusuna ait endpoint yapılandırmalarını içeren dosya.

models: Eğitim sonucunda oluşturulan modellerin bulunduğu dizin.

requirements.txt: Python bağımlılıklarını içeren dosya.

Server: Flask sunucusunun bulunduğu dizin.

flaskServer.py: Flask sunucusunun ana Python dosyası.

1 - Docker Sistemi Üzerinde RASA Tabanlı Bot Çalıştırılması: Dockerfile

Dockerfile, Docker konteynerları oluşturmak için kullanılan bir dosyadır. İçerisinde, bir konteynerın nasıl yapılandırılacağını ve hangi adımların izleneceğini tanımlayan talimatlar bulunur. Öncelikle bir docker build almak için şu şekilde geliştirdiğim Dockerfile'ı açıklayayım:

```
# RASA'nın çalışması için son sürüm imajı temel alıyoruz
# 1
FROM rasa/rasa:latest

# Botun kodlarını kopyalıyoruz konteyner içine (bunu yapabilmek için proje
dizininde bu işlemlerin yapılması gerekiyor)
# 2
COPY . /app

# Çalışması dizini olarak /app'i seçiyoruz, ki sonraki komutlarımız bunun
üzerinde çalışsın
# 3
WORKDIR /app

# ROOT yetkisi ver ve daha sonra sanal ortamdaki dosyaları kullanabilmek için
çalıştırma izni ver
# 4
USER root
# 5
RUN chmod -R 777 /opt/venv/lib/python3.10/site-packages/

# Gerekli bazı bağımlılıkları indiriyoruz
# 6
RUN pip install --no-cache-dir -r requirements.txt
# 7
RUN pip install pandas
```

```
# Expose port (RASA için 5005 portunu default olarak ayarlıyoruz)
# 8
EXPOSE 5005

# RASA çalıştırma komutu
# 9
CMD ["rasa", "run", "--enable-api", "--cors", "*", "--port", "5005"]
```

Line 1:

Bu satır Docker imajının Rasa'nın en son sürümünü temel alarak oluşturulacağını belirtir.

`rasa/rasa:latest` ifadesi, Rasa'nın Docker Hub üzerindeki en güncel sürümünü kullanmamızı sağlar. Bu, Rasa'nın tüm temel bağımlılıkları ve çalışma ortamı ile birlikte indirilmesini sağlar (Yaklaşık 1 GB dosya indirilecektir bu adımda).

Line 2:

Bu komut, bulunduğunuz proje dizinindeki tüm dosyaları (RASA kodları ve yapılandırma dosyaları da dahil) Docker konteynerının içindeki `/app` dizinine kopyalar. Bu komut sayesinde, geliştirdiğim botun tüm kodları konteyner içinde bulunur ve çalıştırılabilir.

Line 3:

Bu komut, Docker konteynerında çalışacak tüm komutların `/app` dizini içinde çalıştırılacağını belirtir. Böylece, sonradan çalıştırılacak komutlar bu dizin içinde işlem yapar (Mesela 9. adımdaki komut bu dizinde çalışacaktır).

Line 4:

Bu komut, konteyner içinde root kullanıcısı olarak işlem yapmamızı sağlar. Root kullanıcısı, sistem üzerinde tam yetkiye sahip olduğu için dosya izinlerini değiştirme ve paket yükleme gibi işlemler yapılabilir. (Normalde bazı işlemler için bu yetki isteniyor, her adımda root şifresi istenmesinin önüne geçmek için Dockerfile içine koydum bunu da)

Line 5:

Dockerfile'ı ilk yazdığımda bu kütüphane ve yapılandırma dosyalarını kullanamıyordum ve `permission denied` hatası alıyordum. Ardından root yetkisi verdiğimde de dosyaların çalıştırılabilir formatta olmadığı hatası almaya başladım. Çözüm olarak (`chmod +x`) ile tüm kütüphane dosyalarına çalıştırılabilme yetkisi verdim. Bu komut, Rasa'nın sanal ortamındaki dosyalara tam erişim izni verir. `chmod -R 777` komutu, belirtilen dizindeki tüm dosya ve klasörlere okuma, yazma ve çalıştırma izinleri verir.

Bu, Rasa'nın bağımlılıklarını düzgün bir şekilde çalıştırabilmesi için gereklidir çünkü bazı arka planda çalışan bağımlılıklar kütüphaneler aracılığıyla bu dosyaları kullanır dosyalar da çalıştırabilme (+x) yetkisine sahip olmadığı için hata alıyordum. Bu durumu bu şekilde düzelttim ancak normalde `venv` ile derleyebildiğim bot; bu şekilde çalıştırınca neden bu yetkiyi kaybetti, net

bir fikrim yok ancak tahminimce `venv`'lerde ve Docker sistemlerinin çalışma farkından dolayı kaynaklanıyor. Docker sistemden bağımsız çalışır bu yüzden olabilir.

Line 6:

Bu komut, `requirements.txt` dosyasındaki bağımlılıkları yükler. `--no-cache-dir` bayrağı, pip'in önbellek kullanmamasını sağlar, bu da daha temiz bir kurulum sağlar.

requirements.txt:

```
rasa
```

Line 7:

Bu komut, Pandas kütüphanesini yükler. Normalde çok şart değil aslında ancak yazdığım botta bu kütüphaneyi kullanmıştım bu yüzden çalışabilmesi için bunu da indirdim. Aynı şekilde sistemde yüklü olmayıp kullanmamız gereken kütüphaneler vs. varsa bu şekilde indirmeliyiz Dockerfile ile.

Line 8:

Bu komut, Docker konteynerının 5005 portunu açar. Bu, Rasa sunucusunun bu port üzerinden erişilebilir olmasını sağlar. Bu port default olarak RASA için çalışmış olacak artık.

Line 9:

Bu komut, Docker konteynerı başlatıldığında çalıştırılacak varsayılan komutu belirler. Burada, Rasa sunucusu `--enable-api` ve `--cors "*" , --port "5005"` seçenekleri ile başlatılır. `--enable-api` Rasa'nın API modunu etkinleştirir, `--cors "*"` tüm kaynaklardan gelen isteklere izin verir ve `--port "5005"` port 5005'te sunucuyu dinlemeye başlar.

Yani bu komut kısaca RASA botunu bir api olarak başlatır. Normalde çalıştığımız gibi terminalden açmaz arayüzü. Botu API olarak kullanabilmek için daha önce yazdığımız mobil program için de bu komutu kullanıyordum .

Not: Bu Dockerfile ``rasa train`` adımını içermez çünkü hali hazırda zaten derleyip oluşturduğum bir model vardı. Bunu istemiyorsak Dockerfile'a 9. stepten hemen önce bu komut eklenebilir.

2 - Docker Sistemi Üzerinde RASA Tabanlı Bot Çalıştırılması: docker-compose.yml

Docker Compose, Docker konteynerlerini tanımlamak ve çalıştırmak için kullanılan bir araçtır. Birden fazla konteynerin birlikte çalışmasını kolaylaştırır ve bu konteynerlerin konfigürasyonlarını tek bir dosya (genellikle docker-compose.yml dosyası) ile yönetmemizi sağlar. Bu sayede, karmaşık uygulamaları tek bir komutla başlatabilir, durdurabilir ve yönetebilirsiniz. Ayrıca kullanıcı oturumlarını yöneteceğimiz Redis yapısını da bu dosyada yapılandıracağız.

- **services** anahtarı, tanımlanan tüm konteyner hizmetlerini içerir. Burada rasa adında bir hizmet tanımlanmıştır.
- **build**: Bu satır, Dockerfile'ın bulunduğu dizini belirtir ve Dockerfile kullanılarak imajın oluşturulacağını söyler. Nokta (.) mevcut dizini ifade eder.
- **ports**: Bu bölüm, konteynerin hangi portları kullanacağını tanımlar. 5005:5005 ifadesi, host makinedeki 5005 portunun konteyner içindeki 5005 portuna yönlendirileceğini belirtir. Bu sayede, Rasa botuna yerel makineden 5005 portu üzerinden erişilebilir.
- **volumes**: Bu bölüm, dosya sistemini bağlamayı tanımlar. ./app ifadesi, yerel dizini (.) konteyner içindeki /app dizinine bağlar. Bu, konteynerin yerel dosyalara erişebilmesini sağlar ve yapılan değişikliklerin anında konteyner içinde yansıtılmasını sağlar.
- **command**: Bu bölüm, konteyner başlatıldığında çalıştırılacak komutu belirtir. Burada, rasa run -enable-api --cors "*" --port "5005" komutu çalıştırılır. Bu komut, Rasa sunucusunu API modunda başlatır, CORS ayarlarını yapar ve 5005 portundan dinlemeye başlar.

3 - Docker Sistemi Üzerinde RASA Tabanlı Bot Çalıştırılması: Docker Build

Tüm bu adımlardan sonra şimdi Docker build'i şu şekilde alacağız:

```
~/frstRasa$ sudo docker-compose up --build
```

Bu komut, docker-compose.yml dosyasında tanımlanan tüm hizmetleri başlatır ve gerekli imajları oluşturur.

sudo: Yüksek ayrıcalıklarla (root yetkileri) komutu çalıştırır. Bu, Docker komutlarının çalıştırılması için gereklidir.

docker-compose: Docker Compose aracı kullanılarak hizmetlerin yönetilmesini sağlar.

up: Tanımlanan tüm hizmetleri başlatır. Eğer imajlar mevcut değilse, önce bu imajları oluşturur.

--build: Tüm hizmetler için imajları yeniden oluşturur. Bu, Dockerfile'da yapılan herhangi bir değişikliğin uygulanmasını sağlar.

Komut, proje dizininde bulunan docker-compose.yml dosyasını ve Dockerfile'ı okuyarak gerekli yapılandırmaları alır. Komut çalışırken, başlatılan konteynerların logları terminalde görüntülenir. Bu loglar, hizmetlerin doğru bir şekilde başlatıldığını ve çalıştığını doğrulamak için kullanılabilir. Aşağıda bunların bazı kısımlarını ekran görüntüsü olarak koyacağım.

Çıktı:

```
ahmete@ahmete-Inspiron-14-5401:~/frstRasa$ sudo docker-compose up --build
Building rasa
DEPRECATED: The legacy builder is deprecated and will be removed in a future release.
             Install the buildx component to build images with BuildKit:
             https://docs.docker.com/go/buildx/

Sending build context to Docker daemon  1.062GB
Step 1/9 : FROM rasa/rasa:latest
latest: Pulling from rasa/rasa
dbf6a9befcde: Pull complete
3885270d692e: Pull complete
6214d5facb77: Pull complete
72e79ac92225: Pull complete
bb0cbea8a793: Pull complete
9e952b6124bb: Pull complete
4f4fb70ef54: Pull complete
6a9541aaa5f0: Pull complete
Digest: sha256:afa5d1026a37fb2ace83eeaa43fcf164e83ee06a70a4b0e5332c374e30f9e886
Status: Downloaded newer image for rasa/rasa:latest
--> 30e28005d7be
Step 2/9 : COPY . /app
--> e560c2ad478a
Step 3/9 : WORKDIR /app
--> Running in 27ce14cbb476
Removing intermediate container 27ce14cbb476
--> 1b768ff8408d
Step 4/9 : USER root
--> Running in b875d00ff8c6
Removing intermediate container b875d00ff8c6
--> 51de5dcd41d7
Step 5/9 : RUN chmod -R 777 /opt/venv/lib/python3.10/site-packages/
--> Running in 029684da9479
Removing intermediate container 029684da9479
--> 0f897fec0b9
Step 6/9 : RUN pip install --no-cache-dir -r requirements.txt
--> Running in f6bf31falde0

Step 6/9 : RUN pip install --no-cache-dir -r requirements.txt
--> Running in f6bf31falde0
Requirement already satisfied: rasa in /opt/venv/lib/python3.10/site-packages (from -r requirements.txt (line 1)) (3.6.0)
Collecting pydantic<1.10.3
  Downloading pydantic-1.10.2-cp310-cp310-manylinux_2_17_x86_64.manylinux2014_x86_64.whl (12.8 MB)
    12.8/12.8 MB 297.1 kB/s eta 0:00:00
Requirement already satisfied: pymongo[srv,tls]<4.4,>=3.8 in /opt/venv/lib/python3.10/site-packages (from rasa->-r requirements.txt (line 1)) (4.3.3)
Requirement already satisfied: pyyaml<6.0,>=5.3.1 in /opt/venv/lib/python3.10/site-packages (from rasa->-r requirements.txt (line 1)) (5.4.1)
Requirement already satisfied: twilio<8.3,>=6.26 in /opt/venv/lib/python3.10/site-packages (from rasa->-r requirements.txt (line 1)) (8.1.0)
Requirement already satisfied: websockets<11.0,>=10.0 in /opt/venv/lib/python3.10/site-packages (from rasa->-r requirements.txt (line 1)) (10.4)
Requirement already satisfied: sanic<21.13,>=21.12 in /opt/venv/lib/python3.10/site-packages (from rasa->-r requirements.txt (line 1)) (21.12.2)
Requirement already satisfied: fbmessenger<6.1.0,>=6.0.0 in /opt/venv/lib/python3.10/site-packages (from rasa->-r requirements.txt (line 1)) (6.0.0)
Requirement already satisfied: redis<5.0,>=4.5.3 in /opt/venv/lib/python3.10/site-packages (from rasa->-r requirements.txt (line 1)) (4.5.4)
Requirement already satisfied: aiohttp<3.7.4.post0,<3.9,>=3.6 in /opt/venv/lib/python3.10/site-packages (from rasa->-r requirements.txt (line 1)) (3.8.4)
Requirement already satisfied: boto3<2.0.0,>=1.26.136 in /opt/venv/lib/python3.10/site-packages (from rasa->-r requirements.txt (line 1)) (1.26.141)
Requirement already satisfied: portalocker<3.0.0,>=2.7.0 in /opt/venv/lib/python3.10/site-packages (from rasa->-r requirements.txt (line 1)) (2.7.0)
Requirement already satisfied: dask<2022.10.2 in /opt/venv/lib/python3.10/site-packages (from rasa->-r requirements.txt (line 1)) (2022.10.2)
Requirement already satisfied: joblib<1.3.0,>=0.15.1 in /opt/venv/lib/python3.10/site-packages (from rasa->-r requirements.txt (line 1)) (1.2.0)
Requirement already satisfied: ruamel.yaml<0.18.0,>=0.16.5 in /opt/venv/lib/python3.10/site-packages (from rasa->-r requirements.txt (line 1)) (0.17.21)
Requirement already satisfied: slack-sdk<4.0.0,>=3.19.2 in /opt/venv/lib/python3.10/site-packages (from rasa->-r requirements.txt (line 1)) (3.20.2)
Requirement already satisfied: tarsafec<0.5,>=0.0.3 in /opt/venv/lib/python3.10/site-packages (from rasa->-r requirements.txt (line 1)) (0.0.4)
Requirement already satisfied: jsonschema<4.18,>=3.2 in /opt/venv/lib/python3.10/site-packages (from rasa->-r requirements.txt (line 1)) (4.17.3)
Requirement already satisfied: pluggy<2.0.0,>=1.0.0 in /opt/venv/lib/python3.10/site-packages (from rasa->-r requirements.txt (line 1)) (1.0.0)
Requirement already satisfied: typing-utils<0.2.0,>=0.1.0 in /opt/venv/lib/python3.10/site-packages (from rasa->-r requirements.txt (line 1)) (0.1.0)
Requirement already satisfied: cloudpickle<2.3,>=1.2 in /opt/venv/lib/python3.10/site-packages (from rasa->-r requirements.txt (line 1)) (2.2.1)
Requirement already satisfied: psycpg2-binary<2.10.0,>=2.8.2 in /opt/venv/lib/python3.10/site-packages (from rasa->-r requirements.txt (line 1)) (2.9.6)
Requirement already satisfied: CacheControl<0.13.0,>=0.12.9 in /opt/venv/lib/python3.10/site-packages (from rasa->-r requirements.txt (line 1)) (0.12.11)
Requirement already satisfied: networkx<2.7,>=2.4 in /opt/venv/lib/python3.10/site-packages (from rasa->-r requirements.txt (line 1)) (2.6.3)
Requirement already satisfied: pytz<2023.0,>=2019.1 in /opt/venv/lib/python3.10/site-packages (from rasa->-r requirements.txt (line 1)) (2022.7.1)
Requirement already satisfied: packaging<21.0,>=20.0 in /opt/venv/lib/python3.10/site-packages (from rasa->-r requirements.txt (line 1)) (20.9)
Requirement already satisfied: numpy<1.25.0,>=1.19.2 in /opt/venv/lib/python3.10/site-packages (from rasa->-r requirements.txt (line 1)) (1.24.2)
Requirement already satisfied: scikit-learn<1.2,>=0.22 in /opt/venv/lib/python3.10/site-packages (from rasa->-r requirements.txt (line 1)) (1.1.3)
Requirement already satisfied: mattermostwrapper<2.3,>=2.2 in /opt/venv/lib/python3.10/site-packages (from rasa->-r requirements.txt (line 1)) (2.2)
Requirement already satisfied: PyJWT[crypto]<3.0.0,>=2.0.0 in /opt/venv/lib/python3.10/site-packages (from rasa->-r requirements.txt (line 1)) (2.6.0)
Requirement already satisfied: randomname<0.2.0,>=0.1.5 in /opt/venv/lib/python3.10/site-packages (from rasa->-r requirements.txt (line 1)) (0.1.5)
Requirement already satisfied: webexteamssdk<1.7.0,>=1.1.1 in /opt/venv/lib/python3.10/site-packages (from rasa->-r requirements.txt (line 1)) (1.6.1)
Requirement already satisfied: tensorflow-text<=2.11.0 in /opt/venv/lib/python3.10/site-packages (from rasa->-r requirements.txt (line 1)) (2.11.0)
Requirement already satisfied: prompt-toolkit<3.0.29,>=3.0 in /opt/venv/lib/python3.10/site-packages (from rasa->-r requirements.txt (line 1)) (3.0.28)
Requirement already satisfied: pykwalify<1.9,>=1.7 in /opt/venv/lib/python3.10/site-packages (from rasa->-r requirements.txt (line 1)) (1.8.0)
Requirement already satisfied: regex<2022.11,>=2020.6 in /opt/venv/lib/python3.10/site-packages (from rasa->-r requirements.txt (line 1)) (2022.10.31)
Requirement already satisfied: ...

Step 7/9 : RUN pip install pandas
--> Running in 42afa561574d
WARNING: The directory '/app/.cache/pip' or its parent directory is not owned or is not writable by the current user. The cache has been disabled. Check the permissions and owner of that directory. If executing pip with sudo, you should use sudo's -H flag.
Collecting pandas
  Downloading pandas-2.2.2-cp310-cp310-manylinux_2_17_x86_64.manylinux2014_x86_64.whl (13.0 MB)
    13.0/13.0 MB 598.6 kB/s eta 0:00:00
Requirement already satisfied: pytz<=2020.1 in /opt/venv/lib/python3.10/site-packages (from pandas) (2022.7.1)
Requirement already satisfied: numpy<=1.22.4 in /opt/venv/lib/python3.10/site-packages (from pandas) (1.24.2)
Requirement already satisfied: tzdata<=2022.7 in /opt/venv/lib/python3.10/site-packages (from pandas) (2023.3)
Requirement already satisfied: python-dateutil<=2.8.2 in /opt/venv/lib/python3.10/site-packages (from pandas) (2.8.2)
Requirement already satisfied: six<=1.5 in /opt/venv/lib/python3.10/site-packages (from python-dateutil<=2.8.2->pandas) (1.16.0)
Installing collected packages: pandas
Successfully installed pandas-2.2.2

[notice] A new release of pip available: 22.3.1 -> 24.2
[notice] To update, run: pip install --upgrade pip
Removing intermediate container 42afa561574d
--> 20589ecf2f59
Step 8/9 : EXPOSE 5005
--> Running in 0b108b6fa3f4
Removing intermediate container 0b108b6fa3f4
--> 5db557674e7
Step 9/9 : CMD ["rasa", "run", "--enable-api", "--cors", "", "--port", "5005"]
--> Running in 99cae60cc587
Removing intermediate container 99cae60cc587
--> a814207713e9
Successfully built a814207713e9
Successfully tagged frstRasa-rasa:latest
```

```
Attaching to frstrasa_rasa_1
rasa_1 | /opt/venv/lib/python3.10/site-packages/rasa/core/tracker_store.py:1042: MovedIn20Warning: Deprecated API features detected! These feature(s) are not compat
ible with SQLAlchemy 2.0. To prevent incompatible upgrades prior to updating applications, ensure requirements files are pinned to "sqlalchemy<2.0". Set environment
variable SQLAlchemy_WARN_20=1 to show all deprecation warnings. Set environment variable SQLAlchemy_SILENCE_UBER_WARNING=1 to silence this message. (Background on S
QLAlchemy 2.0 at: https://sqlalche.me/e/b809)
rasa_1 | Base: DeclarativeMeta = declarative_base()
rasa_1 | /opt/venv/lib/python3.10/site-packages/pkg_resources/_init_.py:121: DeprecationWarning: pkg_resources is deprecated as an API
rasa_1 | warnings.warn("pkg_resources is deprecated as an API", DeprecationWarning)
rasa_1 | /opt/venv/lib/python3.10/site-packages/pkg_resources/_init_.py:2870: DeprecationWarning: Deprecated call to 'pkg_resources.declare_namespace('google')'.
rasa_1 | Implementing implicit namespace packages (as specified in PEP 420) is preferred to 'pkg_resources.declare_namespace'. See https://setuptools.pypa.io/en/lat
est/references/keywords.html#keyword-namespace-packages
rasa_1 | declare_namespace(pkg)
rasa_1 | /opt/venv/lib/python3.10/site-packages/pkg_resources/_init_.py:2870: DeprecationWarning: Deprecated call to 'pkg_resources.declare_namespace('mpl_toolkit
s')'.
rasa_1 | Implementing implicit namespace packages (as specified in PEP 420) is preferred to 'pkg_resources.declare_namespace'. See https://setuptools.pypa.io/en/lat
est/references/keywords.html#keyword-namespace-packages
rasa_1 | declare_namespace(pkg)
rasa_1 | /opt/venv/lib/python3.10/site-packages/pkg_resources/_init_.py:2870: DeprecationWarning: Deprecated call to 'pkg_resources.declare_namespace('ruamel')'.
rasa_1 | Implementing implicit namespace packages (as specified in PEP 420) is preferred to 'pkg_resources.declare_namespace'. See https://setuptools.pypa.io/en/lat
est/references/keywords.html#keyword-namespace-packages
rasa_1 | declare_namespace(pkg)
rasa_1 | /opt/venv/lib/python3.10/site-packages/pkg_resources/_init_.py:2870: DeprecationWarning: Deprecated call to 'pkg_resources.declare_namespace('ruamel.yaml
')'.
rasa_1 | Implementing implicit namespace packages (as specified in PEP 420) is preferred to 'pkg_resources.declare_namespace'. See https://setuptools.pypa.io/en/lat
est/references/keywords.html#keyword-namespace-packages
rasa_1 | declare_namespace(pkg)
rasa_1 | /opt/venv/lib/python3.10/site-packages/sanic_cors/extension.py:39: DeprecationWarning: distutils Version classes are deprecated. Use packaging.version inst
ead.
rasa_1 | SANIC VERSION = LooseVersion(sanic version)
rasa_1 | /opt/venv/lib/python3.10/site-packages/tensorflow/python/framework/dtypes.py:246: DeprecationWarning: 'np.bool8' is a deprecated alias for 'np.bool_'. (De
precated NumPy 1.24)
rasa_1 | np.bool8: (False, True),
rasa_1 | 2024-08-01 11:57:47 INFO root - Starting Rasa server on http://0.0.0.0:5005
rasa_1 | 2024-08-01 11:57:49 INFO rasa.core.processor - Loading model models/20240801-105059-ascent-palette.tar.gz...
rasa_1 | 2024-08-01 11:58:11.293250: E tensorflow/core/framework/node_def_util.cc:675 NodeDef mentions attribute epsilon which is not in the op definition: Op=name
=MklFusedBatchMatMulV2; signature=x:T, y:T, args:num_args*T -> output:T; attr=T:type,allowed={DT_BFLOAT16, DT_FLOAT}; attr=adj_x:bool,default=false; attr=adj_y:bool
,default=false; attr=num_args:int,min=0; attr=fused_ops:list(string),default=[]- This may be expected if your graph generating binary is newer than this binary. Unk
nown attributes will be ignored. NodeDef: ((node rasa sequence layer/text/text_encoder/transformer_encoder_layer/multi_head_attention/add))
rasa_1 | 2024-08-01 11:58:21.917810: E tensorflow/core/framework/node_def_util.cc:675 NodeDef mentions attribute epsilon which is not in the op definition: Op=name
=MklFusedBatchMatMulV2; signature=x:T, y:T, args:num_args*T -> output:T; attr=T:type,allowed={DT_BFLOAT16, DT_FLOAT}; attr=adj_x:bool,default=false; attr=adj_y:bool
```

Bu adımlardan sonra build alınmış olunuyor. Süreç içinde birtakım büyük boyutlu indirme ve büyük dosyaların konteyner içerisine kopyalanma işlemleri olduğu için işlem ilk kez yapıldığında biraz uzun sürebiliyor.

Tüm bu adımlardan sonra RASA botumuz API olarak çalışır ve hazır durumda olmuş oluyor. Bu API yerel ağda 5005 portundan komut dinlemeye hazırdır. Şimdi bu adımda da yazdığım flask tabanlı sunucuyu açıklayacağım.

4 - Flask - Server Üzerinden Dockerla Çalıştırılmış Bot ile İletişime Geçme

Öncelikle bu adım için basit bir flask sunucu-istemci kodu yazdım. Kod şu şekildedir:

```
from flask import Flask, request, jsonify, render_template_string
import requests

# Flask uygulaması oluşturma
app = Flask(__name__)

@app.route('/') # Ana sayfa rotası, HTML formu döndürür
def index():
```

```

return render_template_string('''
    <form action="/bot" method="post">
        <input type="text" name="message">
        <input type="submit">
    </form>
''')

@app.route('/bot', methods=['POST']) # Bot ile iletişime geçme rotası
def get_bot_response():

    # Kullanıcı mesajını formdan al
    user_message = request.form.get('message')

    # Rasa botuna HTTP POST isteği gönder
    response = requests.post(
        'http://localhost:5005/webhooks/rest/webhook',
        json={"sender": "test_user", "message": user_message}
    )

    # Rasa botundan gelen yanıtı JSON formatında döndür
    return jsonify(response.json())

# Flask uygulamasını başlatma
if __name__ == '__main__':

    app.run(port=5000)

```

ardından şu şekilde çalıştırılır:

```
~/frstRasa/Server$ python3 flaskServer.py
```

Kodun Açıklaması:

1 - Gerekli kütüphanelerin import edilmesi:

- **Flask:** Flask web framework'ünü kullanmak için içe aktarılır.
- **request:** HTTP isteklerini işlemek için kullanılır.
- **jsonify:** JSON formatında veri döndürmek için kullanılır.
- **render_template_string:** HTML şablonlarını işlemek için kullanılır.
- **requests:** Diğer web hizmetlerine HTTP istekleri göndermek için kullanılır.

2 - Flask uygulaması oluşturulması:

```
app = Flask(__name__)
```

- Flask uygulaması oluşturulur ve app değişkenine atanır. Bu, web uygulamamızı başlatmak ve yapılandırmak için kullanılır.

3 - Ana Sayfa Rota Tanımlaması:

```
@app.route('/')
def index():
    return render_template_string('''
        <form action="/bot" method="post">
            <input type="text" name="message">
            <input type="submit">
        </form>
    ''')
```

- / rotası tanımlanır. Bu rota, form içeren basit bir HTML sayfası döndürür. (Aşağıya yerelde oluşturulan bu HTML sayfasının görüntüsünü ekleyeceğim)
- render_template_string: HTML formunu doğrudan bir string olarak döndürmek için kullanılır.

4 - Bot İle İletişime Geçme Rota Tanımlaması:

```
@app.route('/bot', methods=['POST'])
def get_bot_response():
    user_message = request.form.get('message')

    response = requests.post(
        'http://localhost:5005/webhooks/rest/webhook',
        json={"sender": "test_user", "message": user_message}
    )

    return jsonify(response.json())
```

- /bot rotası tanımlanır ve POST yöntemi ile çalışır.
- request.form.get('message'): Formdan gelen kullanıcı mesajını alır. (Sayfada kutucuk içine bir şey yazılır ve submit edilir)
- requests.post: Rasa botuna HTTP POST isteği gönderir.
- URL: <http://localhost:5005/webhooks/rest/webhook> (Rasa'nın varsayılan webhook URL'si)
 - Bu kısmı `endpoints.yml` dosyasında zaten önce de şu şekilde tanımlamıştık:

```
action_endpoint:
    url: "http://localhost:5055/webhook"
```

- JSON payload: {"sender": "test_user", "message": user_message} (Kullanıcı mesajı ve gönderen bilgileri)

- response.json(): Rasa botundan gelen yanıt JSON formatında alınır.
- jsonify: Yanıtı JSON formatında döndürür.

5 - Uygulamanın Başlatılması:

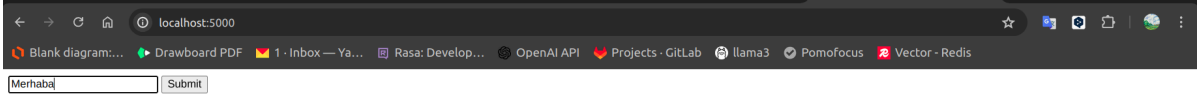
```
if __name__ == '__main__':  
    app.run(port=5000)
```

- Flask uygulamasını 5000 portunda başlatır.
- Bu bölüm, komut dosyası doğrudan çalıştırıldığında çalıştırılmasını sağlar.

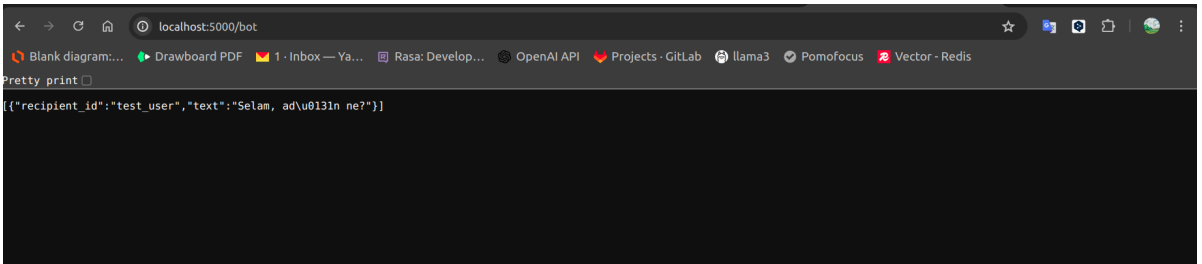
Çıktı:

```
ahmete@ahmete-Inspiron-14-5401:~/frstRasa/Server$ python3 flaskServer.py  
* Serving Flask app 'flaskServer'  
* Debug mode: off  
WARNING: This is a development server. Do not use it in a production deployment. Use a production WSGI server instead.  
* Running on http://127.0.0.1:5000  
Press CTRL+C to quit  
127.0.0.1 - - [01/Aug/2024 15:00:21] "GET / HTTP/1.1" 200 -  
127.0.0.1 - - [01/Aug/2024 15:00:21] "GET /favicon.ico HTTP/1.1" 404 -  
127.0.0.1 - - [01/Aug/2024 15:00:46] "POST /bot HTTP/1.1" 200 -
```

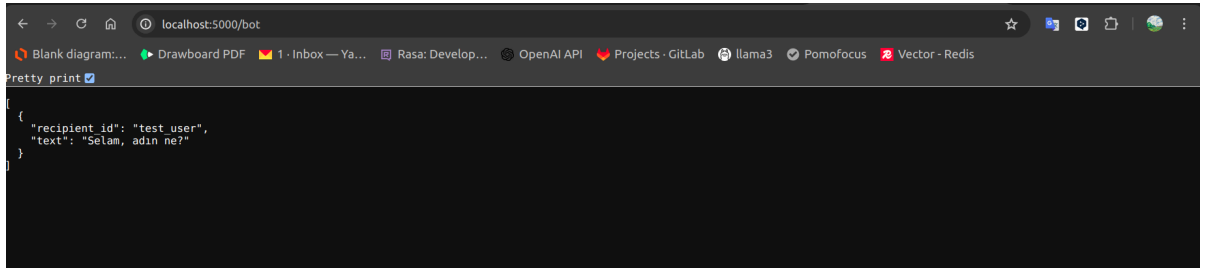
Kodu çalıştırdıktan sonra (<https://127.0.0.1:5000>) adresi ya da diğer bir deyimle (<http://localhost:5000/>) adresi tarayıcıda açılır.



Girilen mesaj submit edildikten sonra cevabın verileceği yeni sayfa açılır.



Pretty print ile daha okunabilir bir şekilde görüntülenebilir (bu kısım daha sonra flask-server dosyasında daha güzel bir görüntü olacak şekilde ele alınabilir);



```
localhost:5000/bot
Blank diagram:... Drawboard PDF 1 · Inbox — Ya... Rasa: Develop... OpenAI API Projects · GitLab llama3 Pomofocus Vector · Redis
Pretty print
{
  "recipient_id": "test_user",
  "text": "Selam, adın ne?"
}
```

Bu raporda, Docker ve Flask kullanarak bir Rasa botunun nasıl konteynerize edildiğini ve API olarak çalıştırıldığını adım adım inceledim. Öncelikle, Dockerfile kullanarak botu bir Docker imajına dönüştürdüm ve ardından docker-compose.yml dosyası ile bu imajı konteynerde çalıştırdım. Son olarak, Flask ile yazdığımız basit bir sunucu aracılığıyla Rasa botuna nasıl istek gönderileceğini gösterdim. Bu süreçte karşılaşılan sorunları ve bunların nasıl çözüldüğünü açıkladım.

Bu çalışmanın sonucunda, Rasa botunu Docker kullanarak izole bir ortamda çalıştırmanın ve Flask sunucusu üzerinden API istekleri alarak botla etkileşime geçmenin mümkün olduğunu gördüm.

Bu rapor, Docker ve Flask kullanarak Rasa botunun nasıl konteynerize edileceği ve API olarak çalıştırılacağı konusunda rehber niteliğindedir.