Chatbot Deployment

Deployment Type:
Chatbot deployment

Technology used:
RASA framework

Language:
Python

Project:
Travelino ChatBot

Project Description:
It is a chat bot application that processes the user information and responds to that , developed based on python RASA framework.

Working of Rasa chatbot:
The chatbot contains 2 parts, rasa server and action server. Action server is responsible for backend requests and Rasa server is the facing server which gets the requests from the application.

Deployment Procedure:

The rasa chat bot deployment contains deployment of two servers, rasa server and action server.

The rasa server is connected to the action server for getting the response.

What is a Virtual Environment?
Python has the virtual environment feature that allows us to use the same package dependencies to run the python code.

The project may contain the pre-built libraries and custom libraries.

For running the chatbot we need to install the dependencies.
We can install the dependencies using requirements.txt file
Or we can use the virtualenv to use the same env to run the chatbot.

Dockerfile for the rasa server

```
FROM rasa/rasa:3.2.6
WORKDIR '/app'

USER root

RUN apt-get update -qq && \
  apt-get install -y --no-install-recommends \
  python3 \
  python3-pip \
  python3-dev \
  # required by psycopg2 at build and runtime
  libpq-dev \
  # required for health check
  curl \
  && apt-get autoremove -y \

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

COPY . /app/rasa_server

RUN . /app/rasa_server/env/Scripts/activate && \
    pip install paginator && pip install pyyaml && \
    pip install SpeechRecognition==3.8.1 && pip install
ffmpeg-python

USER 1001

EXPOSE 5005

CMD
["run","-m","/app/rasa_server/models","--enable-api","--cors","*","--
debug" ,"--endpoints", "endpoints.yml", "--log-file", "out.log",
"--debug"]
```

Here, we use the user root for installing the required packages with root privileges.

And then we installed the “virtualenv using apt” for using the same environment inside the container where the chatbot is created.

<u>NOTE:</u> Virtualenv can be installed by either using apt or using pip but make sure that we install only one way out of two else the conflict will arise.

For activated the existing env folder (where all dependencies are present)

We need to use

```
RUN . /app/actions_server/env/Scripts/activate
```

Where /app/actions_server is the path where code present and /env is the env folder, so we need to give the path of the env to activate it.

From the rasa/rasa image the entrypoint is “rasa” so we are using cmd to append the options to it.
>Rasa run

Dockerfile for the action server

```
FROM rasa/rasa-sdk

WORKDIR /app
USER root
RUN apt-get update && \
    apt-get install  virtualenv -y && \
    virtualenv actions_server

ADD . /app/actions_server/

RUN . /app/actions_server/env/Scripts/activate && \
    pip install paginator && pip install pyyaml

ENTRYPOINT []

RUN cd /app/actions_server/actions/
CMD python -m rasa_sdk --actions actions.actions
EXPOSE 5055
```

Exactly, we need to do the same approach for deploying the action server.
And run the python rasa_sdk module for running the actions with format –actions actions.actions

Here 1st actions is a folder and 2nd actions is actions.py file.

docker-compose.yml

```
version: '3'
services:
  rasa:
   container_name: "rasa_server"
   user: root
   build:
    context: .
    dockerfile: Dockerfile.rasa
   volumes:
   - "./:/app"
   - "./data:/app/data"
   ports:
    - 5005:5005
  action_server:
   container_name: "action_server"
   build:
    context: .
    dockerfile: Dockerfile.actions
   volumes:
    - "./:/app/actions/"
    - "./data:/app/data"
   ports:
    - 5055:5055
```

For intercommunication between the two servers we use the docker compose to launch the servers on the same docker network.

Change the endpoint for the rasa server in the endpoints.yaml
To,

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

Here, action_server is the docker container name of action server.

“Action server” listens on port 5055
“Rasa server” listens on port 5005

We have to expose the respective ports on both server
But we need to bind the rasa server port to the host.

Now, the rasa server is listening on port 5005.

Now configure the reverse proxy on the nginx server to listen on 80 and redirect to 5005.

That’s it, the chatbot server is deployed and live.

For reference.
https://github.com/samik-saha/rasa-chatbot