**Create Server and install MySql on the server**

1. Buy a domain (goldcot.com)
2. a. Create an instance on google cloud

b. Go to Google Cloud > Compute Engine > VM Instances > Create Inctance

c. It opens a new window.

* Enter name : rasa-chatbot-course
* Machine Configuration:
  + Series : N1
  + Machine Type : n1-standard-2 (2 vCPUs, 7.5 GB memory)
* Firewalls : Allow HTTP traffic, Allow HTTPS traffic
* Boot Disk :
  + Operating System : Ubuntu
  + Version: Ubuntu 18.04 LTS
  + Size : 40 GB

d. Click “create”

e. The inctance can be seen on VM Inctances page.

1. Install Putty. **PuttyGen** is also installed automatically in the package. **PuttyGen** will generate SSH key and **Putty** will connect you to server.
2. Run **PuttyGen.** Click **“Generate”.** Move your mouse over grey blank area after you click "generate" button. After completing generation replace  "Key comment" whtever you want. ( We used “**pmc\_rasa**”). Click "**Save Private Key**" to save private key file (“ssh\_private.ppk”) that you will use to connect server. Then copy both the public key and the key comment and add them to google cloud > compute engine>metadata > SSH keys > add item.
3. Run **Putty**.

* Enter Hostname (or IP address) : The External IP from google could VM Instance (34.68.130.89)
* On the left menu > SSH > Auth > Browse : The private key file we saved in the previous step
* Click “Open”. We connected to the server.

1. We are in the server terminal now. We will install MySql in the server.

* Login as : **pmc\_rasa** (This is the username we entered as “username” in google cloud wihle adding SSH key)
* Update server: “ sudo apt update ”
* İnstall mysql : “ sudo apt install mysql-server ”
* Go to mysql : “ sudo mysql ”
* Create a user and Password: “ CREATE USER ‘pmca’@’localhost’ IDENTIFIED BY ‘password123’; “
* Grant this user all privileges:
  + “ GRANT ALL PRIVILEGES ON \*.\* TO 'pmca’@'localhost' WITH GRANT OPTION; ”
  + “ FLUSH PRIVILEGES; ”
* Create a database: “ CREATE DATABASE pmc\_db; ”
* For remote access to database change config file:

1. Go to to directory:
   * + - “ cd /etc”
       - if you can’t find the etc directory: ‘ find / -xdev 2>/dev/null -name "etc" ‘
       - “ cd mysql”
       - “ cd mysql.conf.d ”
2. Install vim editor to open config file
   * + - “ sudo apt install vim ”
3. Update terminal : “ sudo apt-get update “ . run this command every time you install a software!!!
4. Open config file: “ sudo vim mqsqld.cnf”
   * + - Go to the line that begins with the “ bind-address = 127.0.0.1 “
       - Push “insert” key to switch insert mode
       - Edit “#” in the beginning of this line to conver it to a comment
       - Push “escape” to switch command mode
       - Go to the end of the document
       - Write “ :wq! ” to save file and quit
       - if you make a mistake write “:q!” to quit qithout saving
5. restart mysql:

“ sudo systemctl restart mysql“

1. Google Cloud Firewall Settings

* Go to VPC Network > Firewall
* Click “Create Firewall Rule”

1. Name: “rasa”
2. Source IP Ranges : “ 0.0.0.0/0 ” => means all Ips
3. Below specified protocols and ports: click “tcp” and write “ 3306, 5005, 5055” which are mysql port, rasa server port ana action server port respectively.
4. Click “ Create”
5. Wrting tables to database:

* hostname = "34.68.130.89"
* username = "pmca"
* password = "password123"
* database = "pmc\_db"

“””

import pandas as pd

import mysql.connector

from sqlalchemy import create\_engine

# Open database connection

engine = create\_engine('mysql+mysqlconnector://pmca:password123@34.68.130.89:3306/pmc\_db', echo=False)

# write to mysql

df\_employees.to\_sql('employees', con=engine, if\_exists='replace', index = False)

# read from mysql

query = """SELECT \* FROM employees;"""

df\_1 = pd.read\_sql(query, con=engine)

“””

You can use this connection in your “action.py” file wherever needed.

**install Docker on the server**

1. update server: “ sudo apt-get update ”
2. install docker

* “ curl -fsSL https://download.docker.com/linux/ubuntu/gpg | sudo apt-key add “
* ‘ sudo add-apt-repository "deb [arch=amd64] https://download.docker.com/linux/ubuntu $(lsb\_release -cs) stable" ‘

1. Update again : “ sudo apt-get update ”
2. “ apt-cache policy docker-ce ”
3. “ sudo apt-get install -y docker-ce ”
4. Check if docker is running : “ sudo systemctl status docker “
5. We need two docker containers: one for the rasa server, one for the action server. So we need to install docker compose:

* “ sudo curl -L “https://github.com/docker/compose/releases/download/1.24.0/docker-compose-$(uname -s)-$(uname -m)" -o /usr/local/bin/docker-compose “
* “ sudo chmod +x /usr/local/bin/docker-compose “

1. Check docker compose: “ docker-compose --version “

**Deploying Rasa using Docker**

**1.** Create a new folder **: “**PMC\_Deployment“. There will be two folders and two files inside this folder: action(folder), backend(folder), docker-compose.yml, nginx.conf.

2. action(folder) : action.py and Dockerfile. Inside Dockerfile, there are commands like (install mysqlclient, spacy, profanity-filter, etc.) that are needed to run actions.py. You can add more libraries required by actions.py.

3. backend(folder): Allt he other files will be moved to here (nlu, stories, domain, config, etc). There is also a Dockerfile here in which there are commands like installing pip and nltk and running rasa train.

4. docker-compose.yml : used to compose two containers: action and backend

5. nginx.conf : used for handling traffic

6. replace existing “DucklingEntityExtractor” url in the config.yml with the one below:

- url: 'http://duckling.rasa.com:8000'

7. replace existing “action-endpoint” url in endpoints.yml with the one below:

- url: <http://goldcot.com:5055/webhook>

8. create a “credentials.yml” file in the backend folder.

9. Next step is to transfer “ PMC\_Deployment “ folder to the server. To do that we use “Filezilla”.

10. Install “Filezilla“ software.

11. Once you have it installed, click “Open Site Manager” on the top left corner:

- Protocol : SFTP-SSH File Transfer Protocol

- Host: 34.68.130.89 (External IP fom google cloud VM Instance)

- Logon Type (Drop down menu): Key File

- User: pmc\_rasa (username from google cloud SSH key)

- Key File : Browse to the SSH private key file (“ssh\_private.ppk”)

- Click “connect”

- You will see your local computer on the left and the server on the right. You can drag the “PMC\_Deployment” folder and drag it to the right bottom window to move them to the server. In this window you can move new files to the server, move the same files to overwrite existing files in the server or delete files in server.

12. Go to server using Putty, login as “pmc\_rasa”.

13. Go to PMC\_Deployment folder: “ cd PMC\_Deployment ”

14. Build container using docker compose: “ sudo docker-compose up --build ”

15. Go to a browser, enter “ 34.68.130.89:5005 “ . You can see it says “Hello from Rasa”

**Mapping goldcot.com to 34.68.130.89**

1. Go to <https://domains.google.com>
2. Go to DNS > Custom resource records:

* Type : “A” (Default)
* Ipv4 address : 34.68.130.89 (External IP fom google cloud VM Instance)
* Click “add”.
* It says it can take upto 48 hours.
* Go to a browser and enter “**goldcot.com**” if it displays “404 Not Found”, it works. We have nothing on the page. If you enter “**goldcot.com:5005**”, you can see “”Hello from Rasa” if your server is running.

**Adding Free SSL Sertificate using Certbot**

1. We need to shutdown docker server to install “certbot” which provides ssl certificate. Go to server: “ sudo systemctl stop docker ”
2. update server: “ sudo apt-get update ”
3. install certbot : “ sudo apt-get install certbot ”
4. enter “ sudo certbot certonly”
5. domain name: “ goldcot.com ”
6. switch from “pmc\_rasa” to “root” user: “ sudo su “
7. go to “etc” folder : “ cd /etc “
8. go to : “ cd letsencrypt/live/goldcot.com”
9. enter “ls” to see files in the folder. You should see “fullchain.pem” and “privkey.pem” files. You need to copy them under “external\_data” folder in the “PMC\_Deployment/backend” folder. You can get the path of “external\_data” folder via right\_click (Copy urls to clipboard) on the folder on Filezilla. You need to correct the path as below:

* “ cp fullchain.pem /home/pmc\_rasa/PMC\_Deployment/backend/external\_data/fullchain.pem ” .
* “ cp privkey.pem /home/pmc\_rasa/PMC\_Deployment/backend/external\_data/privkey.pem ” .
* You can check on Filazilla if two files are copied under “external\_data” folder.

1. Update Dockerfile under “backend” folder. You need the change the line starts with “CMD”

* Previous version: ‘ CMD [ "run","-m","/app/models","--enable-api","--cors","\*","--debug"] ‘
* New version : ‘ CMD [ "run","-m","/app/models","--enable-api","--cors","\*","--debug","--ssl-keyfile","/app/external\_data/privkey.pem","--ssl-certificate","/app/external\_data/fullchain.pem”] ‘

1. Move the updated “Dockerfile” (under backend folder) to the server via Filezilla. You need to overwrite the older one under the backend folder.
2. Go to “PMC\_Deployment” folder on server: “ cd /home/PMC\_Deployment”
3. Start docker: “ sudo systemctl start docker ”
4. Enable docker : “ sudo systemctl enable docker”
5. Rebuild docker compose: “ sudo docker-compose up --build”. You need to build docker compose whenver you make change files in the “PMC\_Deployment” folder.
6. Go to browser and enter ” https://goldcot.com:5005 “ and you should see it says “Hello from Rasa“.

**Integrating with Facebook Messenger**

1. Go to <https://developers.facebook.com/> > My Apps

2. Click “Create App”

3. On the Pop up:

- Select “Manage Bussiness Integrations”

- App Display Name : “pmcbot”

- App Purpose: Yourself or your own bussiness

4. Go to Messenger > Set up > Access Tokens > Create new page

- Sayfa Adı: PMC Bot

- Kategori: Bir İşletme Değil

- Click “Sayfa Oluştur”

5. Go to Messenger > Set up > Access Tokens > Add or Remove Pages

- Select the page (“PMC Bot”) you have just created in the above step.

- Click “Submit”

6. Go to Messenger > Set up > Access Tokens > Generate Token

- In the Pop up click “Copy”:

EAAFNQH7WhQ0BAOGJZCsPkhcA7RcRGYyMEArkr6DZAtwo1scmRGb37Y1EZCmbbVZBl9HpZCOxWyY3BxRvVtw3HZBDd23rPsxV2uXCMhIJmBoolOtJaj45cbFP8d1gggwJoAW0xByWTZCZAOyLeNC8sSTiRuNVhkA65rywiwz9K3dRW7xGfXzyIkycCDdCXFqLPS4ZD

7. On the left menu, go to Settings > Basic (Be careful, NOT the settings under Prodcut > Messenger)

- Go to App Secret and click “Show”,

- Enter your facebook password

- Copy the App Secret:

0db3edb4b6fa4fb1fd8ca45e6ee938c3

8. Open your “credentials.yml” file in your PMC\_Deployment > backend” folder, write your “app secret” and “token” as below: (just give it a name (“pmc” ) to use later in facebook developer page for “verify” item)

facebook:

verify: "pmc"

secret: "0db3edb4b6fa4fb1fd8ca45e6ee938c3"

page-access-token: "EAAFNQH7WhQ0BAOGJZCsPkhcA7RcRGYyMEArkr6DZAtwo1scmRGb37Y1EZCmbbVZBl9HpZCOxWyY3BxRvVtw3HZBDd23rPsxV2uXCMhIJmBoolOtJaj45cbFP8d1gggwJoAW0xByWTZCZAOyLeNC8sSTiRuNVhkA65rywiwz9K3dRW7xGfXzyIkycCDdCXFqLPS4ZD”

1. Open your “endpoints.yml” file in your PMC\_Deployment > backend” folder, update the url as below:

Previous version:

- url: http://localhost:5055/webhook

Updated version:

- url: http://goldcot.com:5055/webhook

1. Update Dockerfile under “backend” folder. You need the change the line starts with “CMD”

* Previous version: ‘ CMD [ "run","-m","/app/models","--enable-api","--cors","\*","--debug","--ssl-keyfile","/app/external\_data/privkey.pem","--ssl-certificate","/app/external\_data/fullchain.pem”] ‘

Updated Version: CMD [ "run","-m","/app/models","--enable-api","--cors","\*","--debug","--ssl-keyfile","/app/data/external\_data/privkey.pem","--ssl-certificate","/app/data/external\_data/fullchain.pem", "--credentials", "credentials.yml", "--endpoints", "endpoints.yml"]

1. Move the “Dockerfile”, “credentials.yml” and “endpoints.yml” files (under backend folder) to the server via Filezilla. You need to overwrite the older ones under the backend folder.
2. Go to server first stop Docker and and then rebuild docker:

- “sudo docker-compose rm -f”

- “ sudo docker-compose up --build ”

1. When the server is up,

- Go to Messenger > Settings > Webhooks > Add Callback Url

- In th ePop up enter:

. Callback URL: https://goldcot.com:5005/webhooks/facebook/webhook

- Verify Token: pmc (the name we write for verify item in credentials.yml file)

- Click “Verify and Save”

1. Go to Messenger > Settings > Webhooks > Add Subscriptions

- In the Pop up, Select below items and then click “Save”

- messages, messaging\_postbacks, messaging\_optins, messaging\_deliveries, messaging;\_reads, messaging\_payments, messaging-pre\_checkouts, messaging\_checkout\_updates, messaging\_account\_links

15. Now, you can Go to <https://www.messenger.com/> and talk to your bot. Since the bot is in development mode, only you can talk with the bot. You need to register on Facebook to make it public.

16. In the development mode if you want others to use it, Go to <https://developers.facebook.com/> > My Apps > PMC Bot

- On the left menu, go to “Roles”, then click “add tester” and add facebook username the ones who will use the bot. (to find the facebook username go to “Ayarlar ve Gizlilik > Ayarlar” in the dropdown menu on the right top corner). When the one that you add as a tester submit your request, he can talk with the bot.

**DEPLOY RASA CHATBOT TO WEBSITE WITH SOCKET.IO**

1. Add below code to your “credentials.yml” file. Setting “session\_persistence” to True means until you close the browser your conversation stays on the page.

socketio:

user\_message\_evt: user\_uttered

bot\_message\_evt: bot\_uttered

session\_persistence: true

1. Move the “credentials.yml” file (under backend folder) to the server via Filezilla. You need to overwrite the older ones under the backend folder.
2. Go to server first stop Docker and and then rebuild docker:

- “sudo docker-compose rm -f”

- “ sudo docker-compose up --build ”

1. Go to website where you deploy your chatbot. In the html, just before you close <body/> tag, write the below html script that is taken from this github (<https://github.com/botfront/rasa-webchat>)

<div id="webchat"></div>

<script src="https://cdn.jsdelivr.net/npm/rasa-webchat/lib/index.min.js"></script>

// you can add a version tag if you need, e.g for version 0.11.5 https://cdn.jsdelivr.net/npm/rasa-webchat@0.11.5/lib/index.min.js

<script>

WebChat.default.init({

selector: "#webchat",

initPayload: '/get\_started{"userName": "' + userNam + '", "userId": "' + userId + '"}',

customData: {"language": "en"}, // arbitrary custom data. Stay minimal as this will be added to the socket

socketUrl: "https://goldcot.com:5005",

socketPath: "/socket.io/",

title: "PMC",

subtitle: " Your project assitant ",

params: {"storage": "session"} // can be set to "local" or "session". details in storage section.

})

</script>

**How to Do a Clean Restart of a Docker Instance**

1. Stop the container(s) using the following command:

docker-compose down

2. Delete all containers using the following command:

docker rm -f $(docker ps -a -q)

3. Delete all volumes using the following command:

docker volume rm $(docker volume ls -q)

4. Restart the containers using the following command:

docker-compose up -d