# CREATE A CHATBOT IN PYTHON

## PHASE 1 DOCUMENT SUBMISSION

**TEAM MEMBER NAME:Subidsha.S.C**

**311521106100**

**INTRODUCTION:**

As a result of the rapid technological development and the development of the chatbot concept and the time and effort it can save. Many specialized frameworks have emerged to undertake chatbot creation and development. By relying on artificial intelligence, the chatbot has integrated machine learning within it, and it has become more comprehensive and wider for various technological fields. Therefore, we will create a chatbot for the University's Admission and Registration, the project aims to build a chatbot to facilitate the process of accessing information related to students' inquiries towards admissions, Registration and the university itself. The motivation for the work of this project is that there is no university-level equivalent from previous graduation projects, as this project mainly targets all palestinian tawjihi students and other palestinians, non palestinian students . As a conclusion, it lies in answering frequent and common questions by people and providing the answerto these questions at any time the person wants.

**PROBLEM DEFINITION**:

At the start of each academic semester, registration opens for those wishing to join the university in various disciplines, and telephone calls for admission and registration abound. This leads to an increase in the loads and work for the employees of the Deanship of Admission and Registration as a result of the constant pressure of those wishing to register and their families by flocking to the Deanship, so the employees are not able to answer the phone calls and social media. This often leads to many students who wish to register to be ignored.

**DESIGN THINKING**:

We will use the rasa framework to build the required chatbot, but why did we choose rasa ?Will, rasa has a lot of advantages such as:

Highly customisable with various pipelines can be employed to process user dialogues.

- The rasa framework can be run as a simple http server or can be used from python, usingAPIs.
- It has the Rasa-nlu, when run on a server, can mimic other commercial NLP platforms such asLUIS or wit.ai. This makes it easy to migrate an existing application to rasa-nlu.

But as we know, nothing is perfect and rasa has its disadvantages like :

1. server requirements although spacy is a very fast NLP platform, it seems to be very memoryhungry.
2. Learning curve - Installation, configuration and training phases require machine learningexpertise (at least basic level)
3. Context based conversation not available out of the box - Rasa-nlu does not maintain thecontext automatically. This has to be programmed separately into the chat service.

**How To Setup Rasa and start a demo ChatBot[8]**

- First, we have to know that Rasa needs python to work, so we'll install python and the version should be from 3.6 to 3.8 at maximum.

- **Virtual Environment Setup**

  Create and activate the virtual environment using the below commands.

  ```
  python3 -m venv ./venv
  ```

  Activate the virtual environment:

  ```
  .\venv\Scripts\activate
  ```

- Install Rasa open-source with the below command:

**pip install rasa**

- make a directory and move to it

  **mkdir test-chatbot && cd test-chatbot**

  Create the new project with **rasa init** command and start the conversation with the initialdemo chatbot.

  " There are 3 major parts to the Rasa Framework:

  - The Rasa Server: This is the actual software that runs and interprets the user's input and handles the responses based on a trained model (more on that below).
  - The Action Sever: This handles some of the complex form logic if you need it, and/or API calls, etc. This is written in Python, and there are many examples in the Rasa repo to look at. Depending on how complex you need your bot to be, you may or may not need to write anything for this.
  - The chat interface - This is not part of the normal Rasa framework, you'll probably want to grab one of the many chat interfaces, and/or connect Rasa to FaceBook Messenger, Slack, Telegram or other service. Rasa has connectors for these major services to allow you to interact with your bot using commercial chat interfaces."

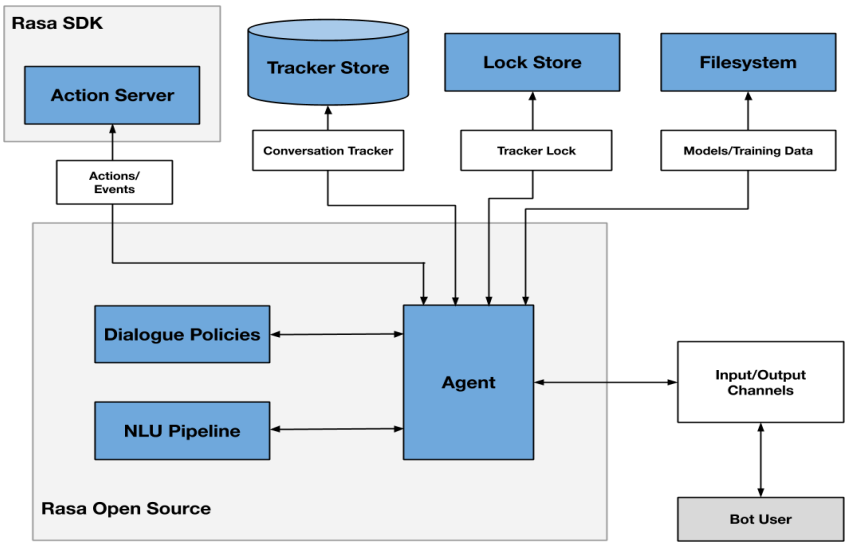
  Training data.

  "The training data for Rasa 2.0 is all formatted as .yml files. There are 3 types of training data.

  - There is the main domain.yml file that contains a list of intents, entities, slots, responses and some actions/form configuration - these equate to what the bot thinks you mean (intent), the text the bot responds with (responses), the bots "memory" (slots and entities), and some setup if you're using the Action server above.
  - The next file is an nlu.yml file - this is your training data, it lists all of your intents and the text, words, phrases that the bot can learn from and interpret as that specific intent. The easiest example is a "greet" intent. That would have things like - hi, hello, how are you, what's up?, how are you doing? as the training data and Rasa will process all that when you train.
  - The last part of the training data is the stories.yml file - these are actual story flows that start with an intent and you script how you want the conversation to flow. These are fluid, they're not super strict "wired up" flows, but they help teach the bot how a conversation should look. If done properly, it will handle tangents (someone talking

about one topic and switching to another and the bot will keep track). You can, and should, have as many stories as possible, different permutations and flows to give Rasa the best idea of a conversation. It'll surprise you with how it works, and will (again if done properly with enough data) converse very well.

To train the bot, it is as simple as typing - rasa train. and getting a cup of coffee or a beer depending on how big your bot is. In the beginning it'll train pretty quickly, but as you add more and more data, it'll take longer and longer. I have one that takes over an hour to train because of the data involved."




-