

Step-by-step guide












- This guide explains how to get started with using the chatbot on a local PC and suggested subsequent steps for using the chatbot on the website. you will go through extracting the zip file, installing the software necessary to run the chatbot, how to run the chatbot and suggested actions for using the chatbot on the Radboud website.
- DISCLAIMER: The software provided here is not a finished product, this is a demo for a chatbot and is not ready for immediate use on the Radboud website.

It is recommended to follow this guide in the order it is provided, for the smoothest installation experience and as not all steps are always required.

Step-by-step guide	1
Downloading and installing the required software	2
Running the chatbot locally	4
Editing the chatbot source code	5
Editing Python	5
Editing Javascript, HTML, CSS	5
Maintenance and hosting of the chatbot	5
Retraining the chatbot	5
Security	7
Hosting	7
Maintenance	8

Downloading and installing the required software

1. Download the zip file we provide, this contains the following:
 - a. the chatbot source code
 - b. The code documentation
 - c. The user manual
2. Extract the zip file
3. Open the folder “team2021-watson”, the content of this folder should look approximately like the following:

 Anonimising script	30/01/2021 14:52
 back-end	30/01/2021 17:18
 Chatbot Workshop	30/01/2021 14:52
 Documentation	30/01/2021 14:52
 front-end	30/01/2021 14:52
 Photoshop	30/01/2021 14:52
 Research	30/01/2021 14:52
 Synonym script	30/01/2021 14:52
 .gitignore	30/01/2021 14:52
 LICENSE	30/01/2021 14:52
 readme	30/01/2021 14:52

4. Install python and pip
 - a. python: go to <https://www.python.org/downloads/> and install the python version corresponding to your operating system **Make sure the python version downloaded is 3.8, 3.7 or 3.6**. Some of the libraries used do not work yet on python 3.9, older versions have not been tested. If the link doesn't work, google “how to install python”.
 - b. pip:
 - i. check if pip is installed:
 1. open a command prompt by typing “cmd” in the windows search bar and click enter
 2. type “pip help”. If list of commands and options shows, then pip is installed, you can skip step ii.
 - ii. Install pip: follow this website:
<https://phoenixnap.com/kb/install-pip-windows>
5. Install the required python packages using pip
 - a. Here are only the QUICK INSTALL instructions:
 - i. Open the folder “team2021-watson”
 - ii. Open the folder “back-end”
 - iii. In the “back-end” folder, left-click with the mouse in the empty space right of the file path as marked with a red cross:

year3 > msdt > team2021-watson > team2021-watson > back-end



- b. type “cmd”, a command prompt will open

- c. type "python setup.py" in the command prompt and click enter. The program will now install the required files and create a shortcut.
 - i. If an error shows mentioning the "c++ redistributable", click on the link in the error message.
 - ii. install the c++ redistributable
 - iii. run the command "python setup.py" again from the commandline as explained in steps a through c.
- d. You have now successfully installed the required files for the chatbot. If something went wrong, refer to the "readme" file in the folder "team2021-watson" and follow the instructions.

Running the chatbot locally

1. When in the previous step the QUICK INSTALL instructions were successfully completed:
 - a. open the runChatbot shortcut on your desktop or runChatbot.bat inside the "team2021-watson" folder. You can skip steps 2 to 4.
 - b. Otherwise continue with step 2:
2. Run the command "python runChatbot.py" in the folder "back-end" inside the "team2021-watson" folder. This can be done with the following steps:

year3 > msdt > team2021-watson > team2021-watson > back-end



- e. type "cmd", a command prompt will open
- f. type "python runChatbot.py" and click enter
- g. A message like below will show, this indicates that the chatbot is running

```
* Serving Flask app "chatbotLocalServer" (lazy loading)
* Environment: production
  WARNING: Do not use the development server in a production environment.
  Use a production WSGI server instead.
* Debug mode: off
* Running on http://127.0.0.1:5000/ (Press CTRL+C to quit)
```

- h. index.html will automatically open showing the chatbot chat window.
3. If running through the cmd doesn't work, run through pycharm
 - a. follow the "Editing python" guide below.
 - b. run runChatbot.py from within the back-end project.
 - c. a message like this will pop up:

```
* Serving Flask app "chatbotLocalServer" (lazy loading)
* Environment: production
  WARNING: Do not use the development server in a production environment.
  Use a production WSGI server instead.
* Debug mode: off
* Running on http://127.0.0.1:5000/ (Press CTRL+C to quit)
```

- d. Index.html will automatically open in a new browser, showing the chat window.
4. If steps 1 till 3 did not work, try the following:
 - a. repeat the steps from steps 2 and 3, but instead of opening runChatbot.py, do the following:
 - b. in the cmd go to "server" inside the "back-end" folder or open chatbotLocalServer.py inside the "server" folder inside the back-end project in your ide.
 - c. run chatbotLocalServer.py
 - d. open the folder "team2021-watson" and then "front-end"
 - e. open "index.html" -> this will open a new browser tab with the chat window

Editing the chatbot source code

Editing Python

1. If you want to edit the Python code, you will need an interpreter. We use PyCharm. Go to <https://www.jetbrains.com/pycharm/download/#section=windows>.
2. Select your operating system, and then download the “community” version.
3. Follow the installation steps.
4. Once installed, open PyCharm and navigate to **file -> open**
 - a. A window will pop up, now go to the “team2021-watson” folder, and open it
 - b. Select the “back-end” folder. Now click OK.
5. You can now edit Python code.

Editing Javascript, HTML, CSS

1. If you want to edit Javascript, HTML and / or CSS code, you will need an interpreter. We use Atom. Go to <https://atom.io/> and download atom.
2. Follow the installation steps.
3. Once installed, open Atom and navigate to **file -> open folder**
 - a. A window will pop up, now go to the “team2021-watson” folder, and open it
 - b. Select the “front-end” folder. Now click “select folder” / “map selecteren”.
4. You can now edit Javascript, HTML or CSS code.

Maintenance and hosting of the chatbot

We suggest a team of software developers to work on the software. These developers should get access to the entire zip-file with code and documentation. The system documentation, which contains explanations about our code and decisions we made, can be found in the “Documentation” folder.

Retraining the chatbot

- The chatbot can learn from user input by letting the user select yes or no after an answer.
 - The question, answer, tag and yes/no response are stored in a csv file called “nn_feedback_data.csv”.
 - **DISCLAIMER:** this is an incomplete feature, below is a suggested guide, but this has not been tested on real user data.
 - Storing the questions and answers could have more purposes than suggested in the guide below.
 -
1. Look through the nn_feedback_data.csv, select the question/answer/tag combinations that seem reasonable.

2. Copy the reasonable question/answer/tag combinations that have a “1” in the “valid” section to the trainingdata.csv
3. retrain the neural network
 - a. In pycharm or your IDE of choice open the “back-end” folder inside the “team2021-watson” folder.
 - b. In your IDE open the chatbot folder
 - c. run the file main.py
 - d. type “1” in the console to train the network.

Security

We have little knowledge about creating applications that are secure for market use. Thus we believe there could be many security gaps in the chatbot application, this should be looked at more closely.

With the little knowledge we have, we would suggest the following actions:

1. Implement security measures:
 - a. For the client-side (Javascript, HTML, CSS)
 - b. And for the server-side (Python)
2. This includes things like client-side input validation (making sure the user can only input certain characters and values), which we have already partly done (with Regex), but mostly, the Python code should be adjusted with security in mind. This is because when Javascript is turned off by the user, all input validation is circumvented.
3. Sending information (the user messages and chatbot answers) between javascript and python should be secure. Currently this information is only passed between javascript and python locally, we do not know the security implications of sending this information over the internet or between servers. We therefore believe the python back-end should be hosted on the same server as the javascript front-end.
4. The chatbot should be kept up to date with the latest security measures.

Hosting

- When is determined that the chatbot is ready for use or for testing on the website, the front-end part of the code should be put up on the University Library website, just like the live-chat. For this, we had already been in contact with Edgar Brendel (e.brendel@ru.nl)
 - 1. Edgar should be contacted and it should be discussed with him what files he needs so that he can put it on the (testing) library website.
 - 2. Then further coordinate with him what needs to happen.
-
- At the moment, the back-end (Python part) is hosted by a local server (a server on the pc of the person testing the chatbot). We do not know the best way to host this back-end server. One idea could be to host the back-end on the same server as the website is hosted, but we do not know how Radboud handles their servers.
 - We, as a team, don't know a lot about servers and how to host code on it. The software developers working on this might know more about it than we do. They should look into it.

Maintenance

- The developers working on this should be reachable through an email address in case anything is wrong with the chatbot. In this scenario, they would fix the error.