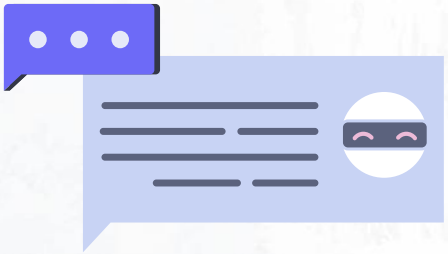
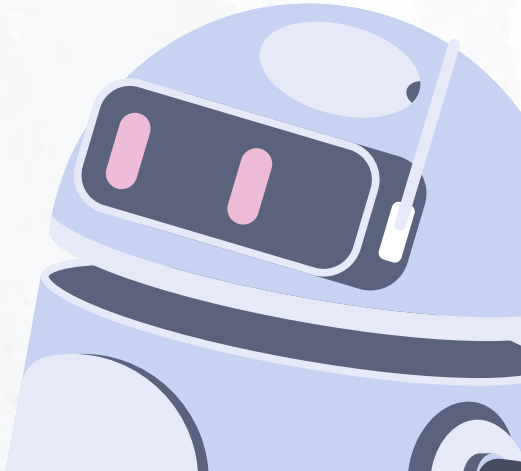
Job Interview Chatbot →

(AI)





Why a Job Interview Chatbot?

Table of contents

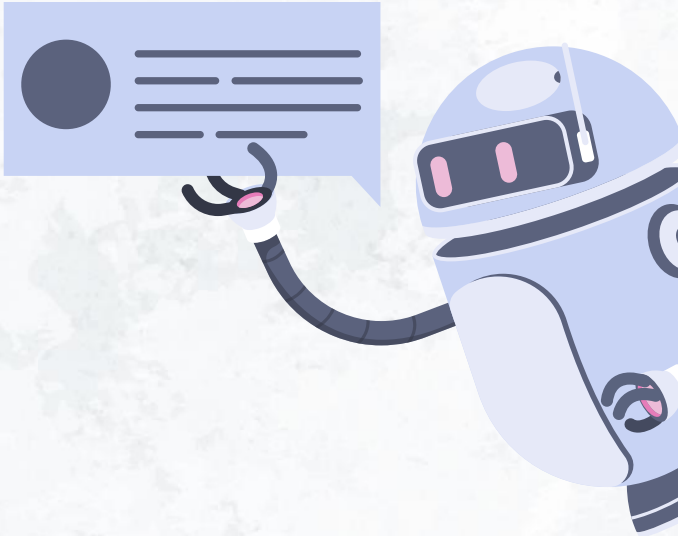
01 →

Which framework was used?

(AI)

(Python)

Rasa →



Why Rasa?

- „Easy“ Setup

Aside from initial installation versioning issues, rasa is decently easy to configure.

- Many Possibilities

Integrations like custom actions allow for many use cases to be realisable.

- Many support platforms

Since rasa is so well established there are many resources which help in learning.

„Easy to use, hard to use right.“

02 →

Thought process behind the use cases

(AI)

Types of interactions

When thinking about the types of interactions the bot will have, there are a few main groups:

(a) General advice ⟶

People come to the chatbot to ask basic questions to prepare themselfs for an interview. This will give general knowledge and help the user prepare and answer their questions.

(b) Learn common interview questions ⟶

Some people might want to look at common interview questions and possibly get advice on how to answer them.

Types of interactions

(c) Mock interviews ⟶

Some users might want to try and answer common interview questions. It would be great if the answers could also be classified and the user can gain feedback for the interview. (Here external resources were needed)

(d) Specific branch questions ⟶

The user might ask for questions in a specific branch. What might be possible questions or further resources to look at? (In this case the scope was limited to only support the specific branch development)

03 →

Implementation of the chatbot

(AI)

Data

34 Intents: 335 Lines of NLU

4 Entities

12 Actions: 647 Lines of Actions

2 Forms

7 Slots

35 Responses

15 Rules: 99 Lines of Rules

58 Stories: 616 Lines of Stories

Parts of the chatbot

(a) Frontend ⟶

The chatbot has a fully functional frontend created in streamlit.

(b) Rasa bot with actions ⟶

The chatbot has a working chatbot server and action server.

(c) Integration with AWS DynamoDB and Lambda ⟶

AWS Lambda and DynamoDB is used to store the job interview questions.

(d) Integration with custom models via Ollama and Transformers ⟶

Rasa can resort to help from Ollama models and Transformers for classification and selected tasks.

(e) Integration with LeetCode via API and GraphQL ⟶

Rasa can retrieve relevant coding questions for the user from LeetCode via GraphQL.

04 →

Live demonstration

(AI)

Try it yourself!

(+) Clone via Git ⟶

git clone --recursive https://github.com/NicoDoebele/job-interview-chatbot.git

(submodules: git submodule init && git submodule update)

(+) Start via docker ⟶

docker compose up -d

(+) Optional: Set up Ollama ⟶

docker exec -it job-interview-chatbot-ollama ollama pull llama3.2

(also found in script: init-ollama.sh)

Thanks! →

CREDITS: This presentation template was created by **Slidesgo** and includes icons by **Flaticon**, infographics & images by **Freepik** and content by **Eliana Delacour**