

Internet Technologies

"Chatbot Project"

Complete **Tasks 1-2** in order to pass the PrL (Praktikumsleistung).

Note the following submission rules:

- Groups of up to 4 students can make a joint submission.
- The submission must include:
 - The names and student IDs of all group members.
 - All source code and configuration files.
 - A setup manual that documents offline installation and usage.
- The submission must not include:
 - Dependencies such as the `node_modules` directory. Instead, provide a `package.json` and `package-lock.json` in the source code.
 - Microsoft Azure account credentials.
 - Any secrets, such as API keys or access tokens.
- Submit the solution in the form of a ZIP archive via iLearn.

Task 1 - Chatbot Implementation

Implement a chatbot, i.e., a computer program that simulates human conversation through text chats.

The following **functional** requirements shall apply:

- The bot should be able to understand and interact within conversations, conducted in Q&A¹ fashion, and limited to a single topic of your choice (e.g, restaurant, airport, gym, etc).
 - The bot should steer the conversion by asking aggressively.
 - The bot should understand the intents from a user's answer (utterance).
 - The bot should have a soft fallback if it cannot understand an user's utterance (i.e., ask the user to rephrase).
 - The bot should have a hard fallback if the bot failed to advance the conversation repeatedly (i.e., start all over again).

¹Question and answer

- The bot should accomplish certain tasks in context of the chosen topic and therefore remember the conversation history. Examples:
 - Place an order in a restaurant.
 - Compile a workout plan in a gym.
- The bot should be able to conduct a conversation that exceeds 20 Q&A turns.
- The bot should not repeat itself (except for fallback questions).
- The user interface should have a modern look & feel supporting responsive web design for mobile devices, tablets, and desktop computers..
- The user interface should display questions and answers alternating in a list-like shape.

The following **non-functional** requirements shall apply:

- The frontend should be implemented in the form of a HTML5 website.
- The frontend should have a layout realized with CSS.
- The frontend should be based on the React framework having at least 4 React components.
- The frontend should communicate with the backend using Socket.IO and the Websocket protocol.
- The backend should be based on Node.js² and use Express.js to serve the application's user interface.
- The backend should identify the intents of a user based on keyword-spotting³.
- The backend should be extensible in a sense that adding new topics / intents / answers does not require changing the source code.

The following rules for **code reuse** shall apply:

- It is allowed to reuse existing layouts, e.g., from *bootsnip*⁴ or from *onaircode*⁵, except the chat layout by Pavel Komiagin⁶.
- It is allowed to reuse the code examples from the lecture, uploaded to iLearn, except the chat layout by Pavel Komiagin.
- It is **not** allowed to reuse code from any other sources.

Hints:

- In order to make the user interface modern looking and responsive, use a framework such as Bootstrap 5.
- Note the quality of your code will be evaluated as well → write clean code.

²Version 16.15.0 LTS

³The identification of keywords in text and/or utterances

⁴<https://bootsnipp.com/tags/chat>

⁵<https://onaircode.com/html-css-chat-box-examples>

⁶<https://bootsnipp.com/snippets/Z1kBn>

Task 2 - Chatbot Deployment

Deploy the chatbot implementation to the Microsoft Azure⁷ cloud.

The following **functional** requirements shall apply:

- The chatbot should be available via a publicly available URL.
- All communication should be encrypted via SSL/TLS.

The following **non-functional** requirements shall apply:

- Enroll for a "Azure for Students"⁸ subscription to get 100\$ of credits.
- To host the chatbot, use one of the following Azure services:
 - Azure App Services (preferred)
 - Azure Virtual Machines

Task 3 - Chatbot Smarten-Up (optional)

Improve your chatbot by replacing the keyword-spotting approach by an implementation based on the Microsoft Azure's *Language Understanding* (LUIS⁹) service to detect the intent from a users utterance. The same functional and non-functional requirements apply as for Task 1, except for keyword spotting.

Submit a fully operational solution in addition to (not 'instead of') the solution of Task 1 to iLearn. However, for Task 2, deploy the solution of Task 3 instead of (not 'in addition to') Task 1.

Note: Successful completion of this task carries a 10% bonus for the exam.

⁷<https://azure.com>

⁸<https://azure.microsoft.com/en-us/free/students/>

⁹<https://luis.ai>