Amir Suleimenov K12247291

PROJECT IN INTELIGENT INFORMATION SYSTEMS

W2024

**CHATBOT AI(LLM) ASSISTANT**

**Project Goal:**

**Develop a intelligent LLM chatbot assistant prototype**

**Measure and collect performance based on human conversations about onboarding process**

**System(Chatbot) Goal:**  **Assist web portal users(npo managers, volunteers):   
Chatbot should consult user, provide information, and help user to use web platform (Onboarding process)**

**Prototyping Scenario:**  **Have conversation as chatbot with predefined role and knowledge specialization**

**Brief description:**

**A web site using django framework and store authorization user profiles in Mongo database with rendering chat page, which is powered configured Llama LLM model, which running through Ollama instance.**

# Requirements

## Summary System Description

(Prototype for) AI-assisted chatbot for Austrian volunteers web portal.

The system based on modelfile role shall help to users by generating answers for users input. The system shall have authorization before accesing chatbot

## Stakeholder Identification

Change to onion model

### Users

1. NPO managers
2. Volunteers

### People affected by the system

1. User and visitors of web-portal. As they will have alternative choice with more variable informative text
2. NPO managers. As they will more easier manage own account
3. Superviser managers. As they will more better understand cycles of volunteering

based on usage

### Managers (those are probably the instructors)

1. Theoretically web masters or Head of web portal department
2. Educationally Professor Birgit Pröll

### Regulators

1. AI Act of European AI Office
2. Austrian municipal office which regulates on local level supervision of AI Act in Upper Austria

**Using the EARS template, document the functional requirements**

**Instructions / Guidelines**

* Req<ID> <optional preconditions> <optional trigger> the <system  
  name> shall <system response>
* Document as many requirements as you desire
  + Not all of them need to be implemented in this course

**List of functional requirements**

Authentification

**FR100:** The system shall allow user create account (registrate) for the purpose of providing conversation for every user

**FR101:** The system shall allow user to his login (authentificate) by entering his password

**FR102:** The system shall allow user change password based on his account

**FR103:** The system shall use Mongo db as DBMS for easier integration with other parts of web portal

Chatbot

**FR200:** The LLM-powered chatbot shall respond to user inputs in natural language.

**FR201:** The chatbot should generate answer for every message from user input

**FR202:** The system shall be able to understand the user's intention from their input.  
**FR203:** The bot shall extract specific information (entities) from the user’s message, such as dates, names, or task information.  
**FR204:** The system shall maintain the context of a conversation across multiple turns, remembering prior user inputs and responses.  
**FR205:** The system should handle ongoing conversations, where it can maintain state and carry information across multiple exchanges.  
**FR206:** The system shall be able to handle misunderstandings, detect when it cannot understand, and ask clarifying questions.

**FR206:** The system should track and update the current conversation state to respond appropriately.  
**FR206:** The system shall decide the next action (asking for more information, providing an answer, executing a task) based on the current state.  
**FR206:** The system shall adapt to individual users by learning preferences over time.

**FR207:** While in a conversation state, the LLM-powered chatbot shall maintain the context of the conversation until the session ends or the context is explicitly reset by the user.

**FR208:** Where possible, the LLM-powered chatbot should personalize responses based on user preferences or past interactions.

**List of nonfunctional requirements**

Chatbot

**NFR 200:** The LLM-powered chatbot’s architecture shall allow for updates and bug fixes to be deployed without downtime.

## Actors

### Primary Actors (who use the system in their daily activities)

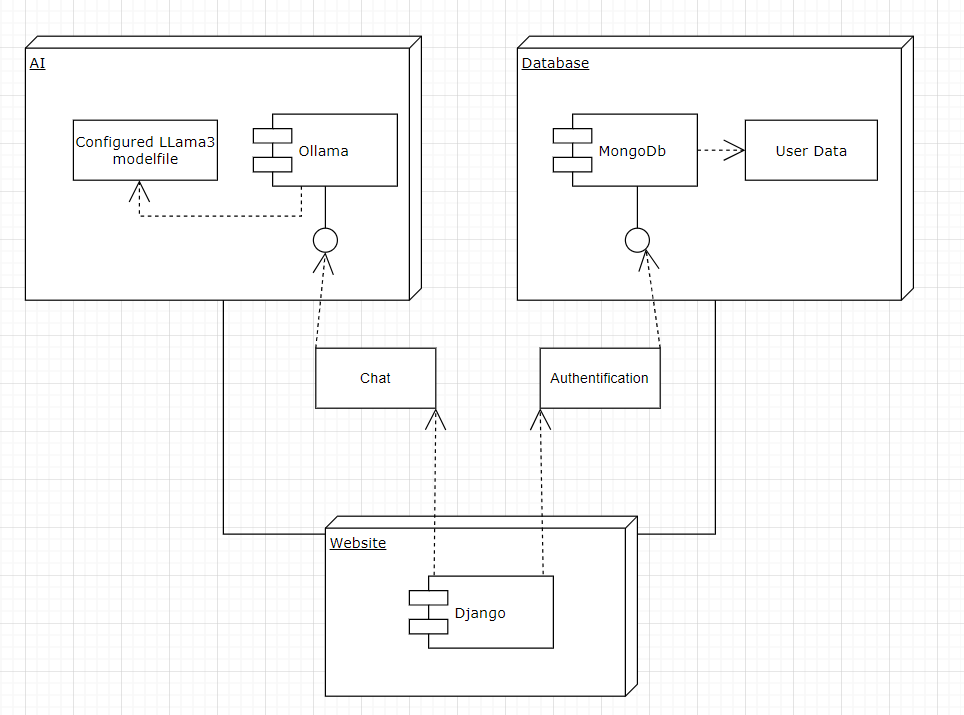
|  |  |  |  |
| --- | --- | --- | --- |
| **#** | **Primary Actor** | **Type** | **Description** |
| 1 | Volunteer/Npo manager as User of chatbot | Human | Responsible for   1. autentification process before starting chat 2. entering messages and questions for defining exact topic of user interest 3. re-stocking shelves |
| 2 | Chatbot Processing Model | AI | Provides core logic for answering on messages in natural language processing   1. Configured as chatbot assistant for Austrian volunteers portal 2. Identifying individual chat and conversation history |

Use Case

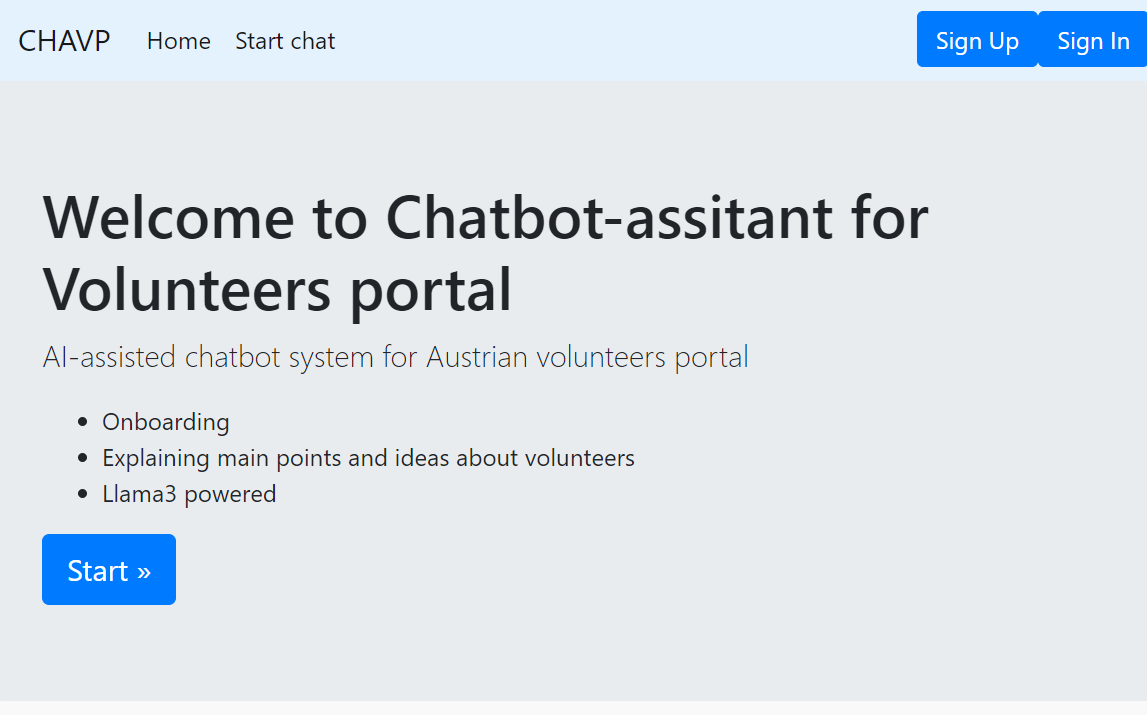
|  |  |
| --- | --- |
| **ID** | UC100-Auth-Login |
| **Description** | User logs into app |
| **Actors** | Volunteer/Npo manager as User of chatbot |
| **Stakeholders:** | NPO managers, Volunteers |
| **Pre-Conditions** | The system has access to the intranet via the device’s WLAN access point |
| **Success end condition:** | The user is logged into the app and into the chatbot in the background |
| **Failure end condition:** | The user is not logged in and is being asked to contact support to verify credentials & permissions |

|  |  |  |
| --- | --- | --- |
| **Main Success Scenario** | | |
| 1 | The user enters his user info (user id, password) | |
| 2 | The user submits the user info | |
| 3 | The systems checks whether the user is permissioned to use the chatbot app | |
| 4 | The system creates a session for the user for submitting API requests LLM engine | |
| 5 | The app launches and displays the main screen/menu | |
|  |  | |
| **Alternative Scenarios** | | |
|  |  | |
| **Exception Scenario** | |  |
| 3.A1.1 | Checking the user credentials results in an authentication error (unknown/invalid user) | |
| 3.A1.2 | The system asks the user the verify credentials and to try to login again | |
| 3.A2.1 | Checking the user credentials results in an authorisation error  (user is known but not permissioned to use the app) | |
| 3.A2.2 | The system informs that permission to use the app is missing and to contact support  Before trying to log in again | |
| 4.A3.1 | The system is unable to create a session with the chatbot system API (either authentication, authorisation or general technical error) | |
| 4.A3.2 | The system asks the user to verify network connection and permissions  Inside the inventory management system before trying to log in again | |

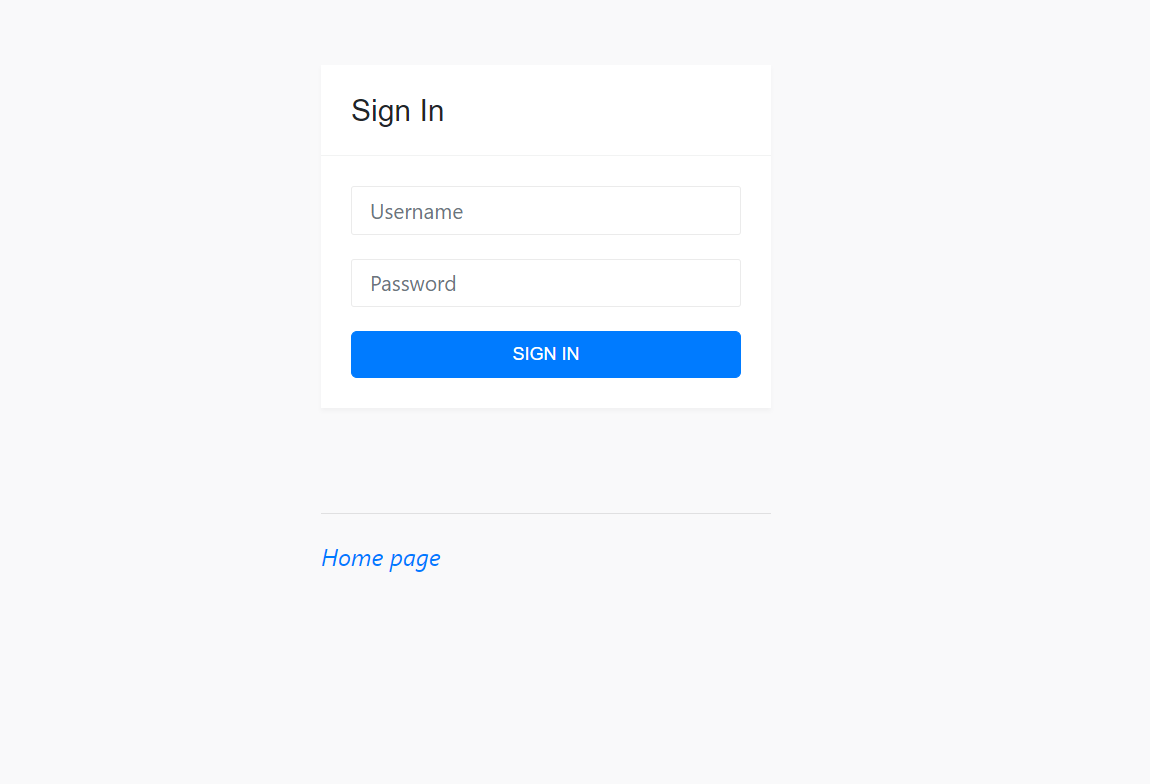
Base architecture Diagram



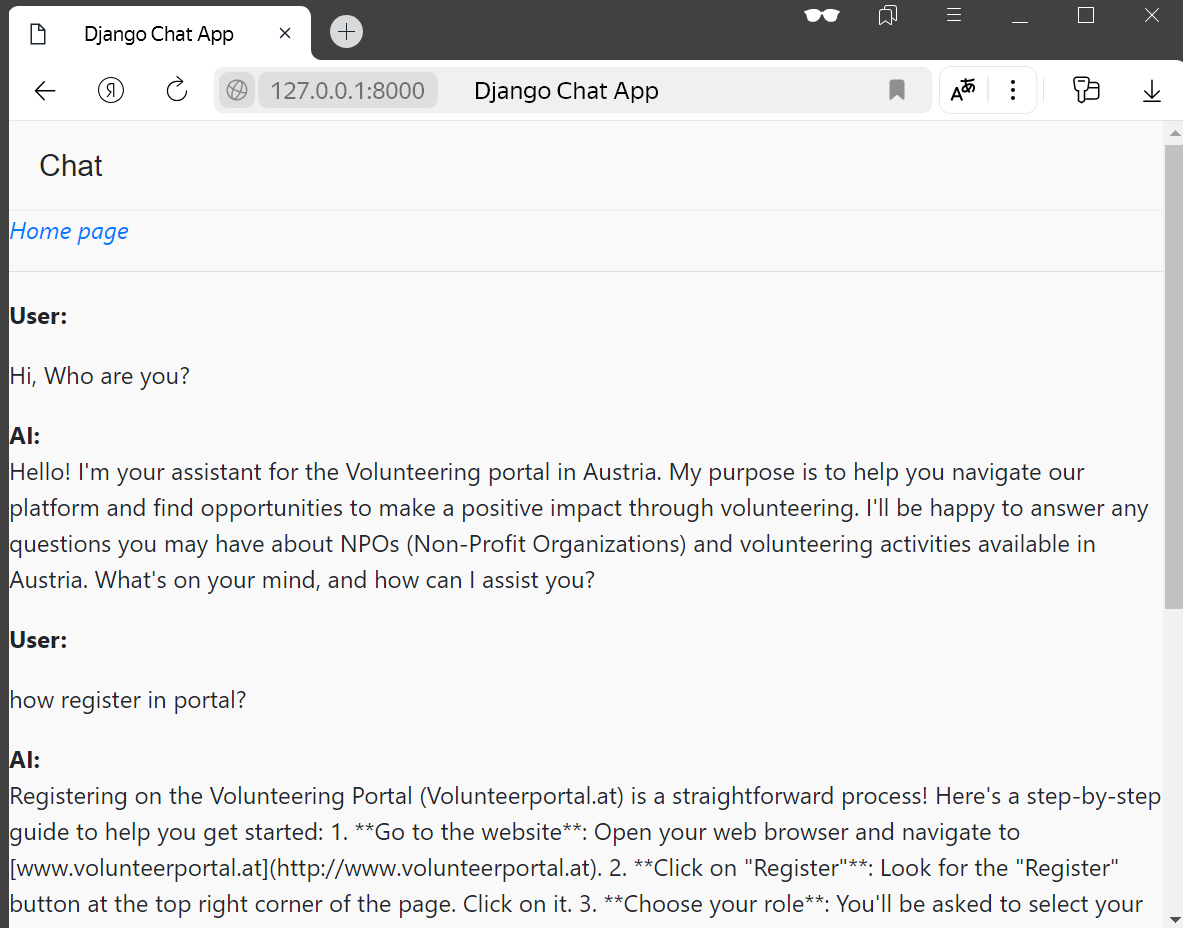
Screenshots examples



main page



login page



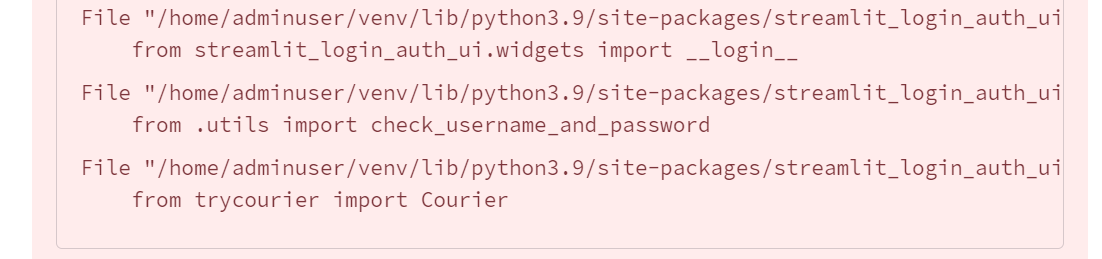
chat

So now chatbot can answer questions and even try to generate close to reality answer when tries to answer as example register procedure.

Future steps and troubles.

For next steps of developing it could be adding difference on behavior between volunteer and npo manager like concentrating on different topics and themes.

To create task creation procedure I faced with some restrictions of LLMs as variability of answering of same question. So to create working pocedure there is need of rule based approach as in usual chatbot with button and questions with strict restrictions of rules like format of enter and etc to future conversion to JSON or XML format for sharing it to other services.

Also I tried to set up chatbot with streamlit instead django. Despite the main indicator as time for response from LLM is was reduced from 10-15 seconds to 4-8 seconds, I faced with autorization problems as it had some troubles in connecting to Mongo DB. When I tried to use predefined authorization schemes with 3rd party applications or libraries(like Courier and Telegram) but they had internal problems as on the image. Same situation also with demo apps which use them on streamlit hosts.

Could you please recommend what future steps I can add to project from this level to improve quality? And how to solve troubles which was listed before?