Assignment 2 - Deliver the requirements of your system

k12104186 Leon Mandler

k12102749 Sandro Assek

Part 1: Describe the goals of your system

- Non-AI goals:
  Provide a more immersive way to interact with villagers. Add a new gameplay element to Minecraft.
- AI related goals
  Give the villagers a unique personality, and make it feel like they are alive.
    - For these, provide one or more reasons why you need AI:
        - Pre-written standard responses wouldn't be very fun/interactive after a while
        - Having a LLM to generate custom responses, is the best way to make the interactions with NPCs (trading) more real-life-friendlier

Part 2: Identify the stakeholders

- Users:
  People who want to enhance their gameplay with mods.
- People affected by the system:
  All the people who use the mod, including developers.
- Managers (those are probably the instructors):
  "Probably the instructors"
- Regulators:
  The Minecraft developers might be concerned with having an LLM in the game. (Potential Jailbreaking of the LLM, possibly making the game less child friendly)

Part 3 Using the EARS template, document the functional requirements

- **ReqCorrectModLoading:** IF the mod is installed when the game is opened the Fabric Mod Loader shall inject the mod into the game
- **ReqGUIdrawing:** IF its inventory is openable WHEN a villager is right clicked the screen class shall add its GUI elements to the screen.
- **ReqWritingFieldFocusing:** IF the writing field is unfocused WHEN the mouse clicks on the writing field or enter is typed, the screen shall set the writing field to focused.

- **ReqMessageSending:** IF the writing field is focused WHEN enter is typed, the Network Handler shall send the prompt packet to the server and the GUI shall clear the writing field and add the prompt to the chat window.
- **ReqResponseParsing:** IF the LLM response contains commands WHEN the response is received on the server, the server side mod shall change the villagers prices.
- **ReqConversationSaving:** IF the player had a previous interaction with the villager, WHEN the villager is opened the screen shall display the previous conversation.

Part 4: Using the EARS template, document the non-functional requirements

- External Interfaces:
  - **NFReqAPIResponse:** When the prompt is received on the server the API interface shall make an API request to a LLM API.
- Performance:
  - **NFReqResTime:** If the internet connection is good and the API operates normally when the LLM receives the prompt, the LLM shall respond within 3 seconds.
  - **NFReqVillagerInteraction:** While it is possible to trade with villagers, the user shall be allowed to interact with villagers.
  - **NFReqChatMessages:** While the villager is alive and starts a conversation all previous chat messages shall be visible to the client.
  - **NFReqOutputQuality:** When an API request is made the mod shall provide a clear role for the LLM, that leads to quality responses.
- Attributes:
  - **NFReqNewGameVersion:** When a new version of the game is released, the mod shall be easily portable to the new version.
  - **NFReqDocumentation:** While the source code is maintained by developers, the source code shall offer a full and up-to-date documentation.
  - **NFReqNewFunctionality:** When adding new functionalities, the new code shall be easy to integrate to existing code.
- Constraints:
  - **NFReqRunEnv:** While minecraft is running on Java, the mod shall run on java.
  - **NFReqGameStandards:** While the fabric api gets updated, the mod shall make good use of it and adhere to its standards
  - **NFReqRunOnJavaOnly:** While there is no good modding tool for other versions, the mod shall only be available in Java edition.
  - **NFReqCleanShutdown:** When the game is closed or the server is shut down, the database shall have the conversations stored.

Part 5: Indicate AI-related requirements (use their IDs)

NFReqOutputQuality: The prompt, that describes its role to the language model, must be well designed to enforce the quality of the output. The responses should sound like they come from a Minecraft villager, and their style should mimic that. The prompt must be fine tuned until the output meets those standards. There must be a balance of how hard it is to convince the villager to change its prices. If it's too easy, the game isn’t fun anymore, and if it’s too hard, the mod serves little purpose. The prompt should contain sentences that prohibit the LLM from sharing its system instructions and changing them. The starting prompt should contain various dynamic parameters that differ from villager to villager, so they all have different talking styles and different backstories. (age, profession, biome, etc.)