Step 1 – Define the set of possible roles an agent may assume

**Output:** A set of roles used in your program with a description of each role  
**Description:** Step 1 requires the developer to identify the set of roles an agent may assume in the desired application. Possible roles include things like teacher, student, parent, child, and coworker. A single agent (in our case user) can assume multiple roles and a role can be restricted to apply to the agent only if certain criteria are met. For example, the role of coworker may only apply if the two agents communicating work for the same company.  
**HINT:** There are at least two.

Step 2 – Describe the expressives exchanged between roles

**Output:** A set of expressive messages with a description of each message  
**Description:** Step 2 requires the developer to describe the expressive messages or expressives exchanged between various roles. Searle defines expressives as communicative acts that enable a speaker to express their attitudes and emotions towards a proposition. Examples include statements like “Congratulations on winning the prize!” where the attitude and emotion is congratulatory and the proposition is winning the prize. Formally, the structure of an expressive is...

\[
\langle \text{sender, receiver, type, proposition} \rangle
\]

The type of the expressive refers to the attitude and emotion of the expressive and the proposition to its content, including the relevant events. The sender and receiver are selected from the set of roles defined in Step 1. Formulate the expressives that can be exchanged among agents assuming those roles. The result is a set of all valid expressive messages allowed by the application.

Step 3 – Derive the emotions to be modeled from the expressives

**Output:** A set of emotions with a description of how each emotion was selected  
**Description:** Step 3 requires the developer to select a set of emotions to be modeled from the emotion ontology (Fig. 1). The selected emotions are based on the expressives identified in the previous step. To compute the set of emotions, we evaluate each expressive and select the most relevant emotions from the ontology for that particular expressive. The selected emotions are then added to the set of emotions required by the application. This process is repeated for every expressive and the resulting emotion set is the output of this step.
The recommended approach to selecting an emotion is to structure Elliot's set of emotions as a tree (Fig. 1). Each leaf of the tree represents two emotions, one that carries a positive connotation and the other a negative connotation. Given an expressive, you start at the top of the tree and using its type and proposition you filter down through the appropriate branches until you are left with only the applicable emotions. For example, say that you have a message with a type of excited and a proposition equal to “I won the game”. Now using the tree you determine that winning the game is an action the user would have taken and that excited has a positive connotation, so the applicable emotion must therefore be pride. In general, the sender and receiver would have different interpretations. For example, if the recipient of the above message is the agent who lost the game, then the emotions that are relevant to the recipient would be admiration and reproach depending on their perception of the winner.

If the proposition of the expressive message is a composite or even ambiguous as to whether or not the type applies to an event, action, or object then more than one path of the tree may apply. Such is the case when an agent conveys its mood via an expressive message. Mood is an aggregation of emotions and therefore does not have a unique causal attribution. For example, an expressive might convey that a user is generally happy or sad without being happy or sad at something. Therefore, we do not select any specific emotion when evaluating an expressive pertaining to mood as the emotions that comprise the mood are captured when evaluating the other expressives.