21162101012_CBA_Yash_Lakhtariya

AI Practical 2

AIM - Develop the following two intelligent agents using the language of your choice. Define PEAS for both of them.

1. Create a facial gesture selection Agent.

Objective -

- We will produce gestures such as smiling, blinking, and nodding to make the agent seem more realistic.
- Program the gesture selection component where the agent perform a particular action based on two elements,

```
The answer element, one of positive, negative, unsure
The mood element, one of: happy, sad, neutral
```

- The agent's gesture selection strategy will be as follows:

```
If happy and either positive or unsure, smile
If happy and negative, nod
If sad, frown
If neutral and positive or negative, nod
If neutral and unsure, blink
```

- **Sample Input - ("happy", "negative") (tuple)**
- **Sample Output - Perform nodding gesture**
- **Test Input - ("neutral", "unsure"), ("sad", "unsure")**

```
[1] !pip install experta
```

```
from experta import *
from IPython.display import Image, display

class ans(Fact):
    pass

class mood(Fact):
    pass

class AIGesture(KnowledgeEngine):
    @Rule(AND(ans(gs=L("positive") | L("unsure")), mood(gs="happy")))
    def smile_or_nod(self):
        display(Image(url="https://media.tenor.com/OWvn6v0O3ssAAAAi/smile-sarcastic.gif"))

    @Rule(AND(ans(gs=L("positive") | L("negative")), mood(gs="neutral")))
    def nodding(self):
        display(Image(url="https://media1.tenor.com/m/U10ApTThsGkAAAAC/skype-speechless.gif"))

    @Rule(AND(ans(gs="unsure"), mood(gs="neutral")))
    def blink(self):
        display(Image(url="https://media1.tenor.com/m/KXH804YS0w8AAAAC/meme-thinking.gif"))

    @Rule(AND(ans(gs="negative"), mood(gs="happy")))
    def nod(self):
        display(Image(url="https://media1.tenor.com/m/U10ApTThsGkAAAAC/skype-speechless.gif"))

    @Rule(AND(ans(gs="negative"), mood(gs="sad")))
    def frown(self):
        display(Image(url="https://media1.tenor.com/m/tTaCJqHvuLwAAAAC/pleading-puppy-eyes.gif"))

    @Rule(AND(ans(gs="unsure"), mood(gs="sad")))
    def frown(self):
        display(Image(url="https://media1.tenor.com/m/tTaCJqHvuLwAAAAC/pleading-puppy-eyes.gif"))

    @Rule(AND(ans(gs="positive"), mood(gs="sad")))
    def frown(self):
        display(Image(url="https://media1.tenor.com/m/KXH804YS0w8AAAAC/meme-thinking.gif"))

mood_input = input("Enter mood (happy, sad, neutral): ")
ans_input = input("Enter answer (positive, negative, unsure): ")

agent = AIGesture()
agent.reset()
agent.declare(mood(gs=mood_input))
agent.declare(ans(gs=ans_input))
agent.run()
```

```
Enter mood (happy, sad, neutral): happy
Enter answer (positive, negative, unsure): positive
```

2. Create a Respiratory illness classification agent.

- The system should request the user to enter the symptoms as input.
- Based on user input, the system should be able to classify if it is normal flu or COVID19 and if COVID19 then whether it is mild or extreme.
- If the user enters less than 3 symptoms, it should ask to enter more symptoms to diagnose their state.
- It should compare the flu and COVID19 counter to print the diagnosis result.
- Add the link from where you are taking symptoms of COVID and flu.

```
from experta import *

class Diagnosis(Fact):
    pass

class Respiratory(KnowledgeEngine):
    symptoms_weights = {
        'Fever': 3,
        'Cough': 2,
        'Fatigue': 1,
        'Shortness of breath': 3,
        'Respiratory issues': 2,
        'Body aches': 1,
        'Headache': 1,
        'Runny nose': 1,
        'Sore throat': 1
    }

    @Rule(Diagnosis(covid=True))
    def covid_diagnosis(self):
        print("\nYou might have COVID-19 symptoms. Please consult a healthcare professional.")

    @Rule(Diagnosis(covid=False))
    def not_covid_diagnosis(self):
        print("\nYour symptoms don't indicate COVID-19. Please monitor your health and consult a healthcare professional if needed.")

    def calculate_diagnosis(self, symptoms):
        total_weight = sum(self.symptoms_weights[symptom] for symptom in symptoms)
        if total_weight >= 10:  # Adjust this threshold as needed
            self.declare(Diagnosis(covid=True))
        else:
            self.declare(Diagnosis(covid=False))

print("\n\tYSL Respiratory Illness Checker\n")

print("\nPlease answer with y (yes) or n (no):\n")
symptoms = [
    input("Fever? "),
    input("Cough? "),
    input("Fatigue? "),
    input("Shortness of breath? "),
    input("Respiratory issues? "),
    input("Body aches? "),
    input("Headache? "),
    input("Runny nose? "),
    input("Sore throat? ")
]

engine = Respiratory()
engine.reset()
engine.calculate_diagnosis([symptom.capitalize() for symptom, answer in zip(["Fever", "Cough", "Fatigue", "Shortness of breath", "Respiratory issues", "Body aches", "Headache", "Runny nose", "Sore throat"
engine.run()
```

```
        YSL Respiratory Illness Checker


Please answer with y (yes) or n (no):

Fever? y
Cough? y
Fatigue? y
Shortness of breath? y
Respiratory issues? n
Body aches? n
Headache? y
Runny nose? y
Sore throat? y

You might have COVID-19 symptoms. Please consult a healthcare professional.
```