**MINI PROJECT**

**NAME :** DUDHE ANIL AJINATH

**ROLL NO :** TEAD21269

**SUBJECT :** AI

**CLASS :** TE

**BRANCH :** AI&DS

**TITLE :**

**Implement any one of the following Expert System**

**• Information management**

**• Hospitals and medical facilities**

**• Help desks management**

**• Employee performance evaluation**

**• Stock market trading**

**• Airline scheduling and cargo schedules**

**CODE:**

**Hospitals and medical facilities:**

class MedicalExpertSystem:

def \_\_init\_\_(self):

self.knowledge\_base = {

"Headache": ["Rest", "Drink water", "Take over-the-counter pain reliever"],

"Fever": ["Rest", "Stay hydrated", "Take fever-reducing medication"],

"Cough": ["Stay hydrated", "Use cough drops", "Consult a doctor if severe"],

"Sore Throat": ["Gargle with warm saltwater", "Use throat lozenges", "Consult a doctor if persistent"],

"Stomachache": ["Avoid heavy meals", "Take over-the-counter antacids", "Consult a doctor if severe"]

}

def diagnose(self, symptoms):

recommendations = []

for symptom in symptoms:

if symptom in self.knowledge\_base:

recommendations.extend(self.knowledge\_base[symptom])

else:

recommendations.append(f"Unknown symptom: {symptom}")

return recommendations

def main():

print("Medical Expert System")

expert\_system = MedicalExpertSystem()

symptoms = input("Enter the symptoms (comma-separated): ").split(",")

recommendations = expert\_system.diagnose(symptoms)

print("\nRecommendations:")

for recommendation in recommendations:

print("- " + recommendation)

if \_\_name\_\_ == "\_\_main\_\_":

main()

class MedicalExpertSystem:

def \_\_init\_\_(self):

self.knowledge\_base = {

"Headache": ["Rest", "Drink water", "Take over-the-counter pain reliever"],

"Fever": ["Rest", "Stay hydrated", "Take fever-reducing medication"],

"Cough": ["Stay hydrated", "Use cough drops", "Consult a doctor if severe"],

"Sore Throat": ["Gargle with warm saltwater", "Use throat lozenges", "Consult a doctor if persistent"],

"Stomachache": ["Avoid heavy meals", "Take over-the-counter antacids", "Consult a doctor if severe"]

}

def diagnose(self, symptoms):

recommendations = []

for symptom in symptoms:

if symptom in self.knowledge\_base:

recommendations.extend(self.knowledge\_base[symptom])

else:

recommendations.append(f"Unknown symptom: {symptom}")

return recommendations

def main():

print("Medical Expert System")

expert\_system = MedicalExpertSystem()

symptoms = input("Enter the symptoms (comma-separated): ").split(",")

recommendations = expert\_system.diagnose(symptoms)

print("\nRecommendations:")

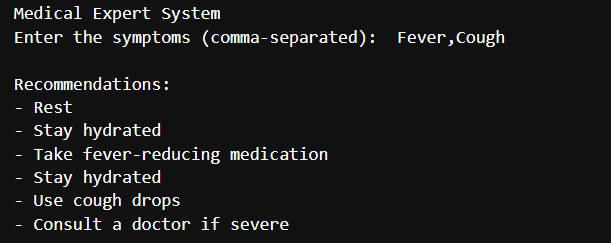
for recommendation in recommendations:

print("- " + recommendation)

if \_\_name\_\_ == "\_\_main\_\_":

main()

**OUTPUT:**

****