Expert system (info system)

# Simple Expert System for Information Management

def expert\_system():

print("=== Expert System: Information Management Assistant ===")

knowledge\_base = {

"data backup": "Ensure regular backups are taken and stored securely.",

"data security": "Use encryption and strong access controls to protect information.",

"storage": "Use cloud storage or databases with redundancy for safe data management.",

"access": "Define user roles and permissions for safe data access.",

"update": "Keep systems and databases updated to avoid vulnerabilities."

}

while True:

query = input("\nAsk about information management (or type 'exit'): ").lower()

if query == "exit":

print("Thank you for using the Information Management Expert System!")

break

elif query in knowledge\_base:

print("System Advice:", knowledge\_base[query])

else:

print("Sorry, I don't have advice on that topic. Try asking about: data backup, data security, storage, access, update.")

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

expert\_system()

theory

What is Information Management in Expert Systems?

Information Management refers to how an expert system organizes, stores, retrieves, and uses data or knowledge for decision-making like a human expert.

🧠 Key Components:

Knowledge Base:

Stores expert rules, facts, and solutions.

Example: "data backup" → "Take regular backups."

Inference Engine:

Applies logical rules to data.

Decides the next action based on input.

User Interface:

Allows users to interact with the system.

Input questions, receive expert suggestions.

🔍 Why It’s Important:

Ensures accurate, fast, and reliable expert decisions.

Helps in data security, backup planning, and efficient access.

Reduces manual dependency on human experts.