#Source code:

```python
#pylint: disable=missing-module-docstring, missing-function-docstring, invalid-name
from dotenv import load_dotenv
from openai import OpenAI
from task_label import TaskLabelPrompt
import json

load_dotenv()
openai_client = OpenAI()

ollama_client = OpenAI(
  base_url="http://localhost:11434/v1",
  api_key="ollama"
)

def get_response_from_openai(user_message: str) -> str:
  response = openai_client.chat.completions.create(
    model="gpt-4o-mini",
    messages=[
      {
        "role": "system",
        "content": "You are a helpful assistant."
      },
      {
        "role": "user",
        "content": user_message
      }
    ]
```

```
  )
  return response.choices[0].message.content


def get_response_from_ollama(user_message: str) -> str:
  response = ollama_client.chat.completions.create(
    model="llama3.1:8b",
    messages=[
      {
        "role": "system",
        "content": "You are technical expert, you determine whether something is a technical problem or not. You answear only with one of two allowed responses: ""problem"" or ""not problem"" You can only repond with either ""problem"" or ""not problem"", without any additional characters."},
      {
        "role": "user",
        "content": user_message
      }
    ]
  )
  return response.choices[0].message.content


def read_json_file(file_path):
  #Reads a JSON file and returns its content as a dictionary.
  with open(file_path, 'r') as file:
    data = json.load(file)
  return data
# ---------------------------------------------------------------------------
# main
# ---------------------------------------------------------------------------
```

```
def main():
    input_json = "C:\\Users\\menmi\\OneDrive\\Dokumenty\\AIAJ\\ai_inventiveness\\reports\\LLMAPI\\classification_task.json"
    data = read_json_file(input_json)
    for x in data:
        message = x["description"]
        x["Ollama_Response"]=get_response_from_ollama(message)
        print(x["Ollama_Response"])
    with open(input_json.replace(".json","") + "_appended.json", 'x') as f:
        json.dump(data, f, indent=4)


if __name__ == "__main__":
    main()



#json output:
```

```
[
    {
        "id": "1",
        "description": "The robotic arm frequently overshoots its position when assembling small components, reducing accuracy.",
        "Ollama_Response": "problem"
    },
    {
        "id": "2",
        "description": "The cooling system in the high-speed engine fails to prevent overheating during prolonged operation.",
        "Ollama_Response": "problem"
```

},
{
"id": "3",
"description": "The drone\u2019s camera stabilizer produces blurry images in windy conditions, impacting video quality.",
"Ollama_Response": "problem"
},
{
"id": "4",
"description": "The packaging machine jams when handling biodegradable materials, causing frequent production stops.",
"Ollama_Response": "problem"
},
{
"id": "5",
"description": "The conveyor belt motor overheats when running at maximum speed for extended periods.",
"Ollama_Response": "problem"
},
{
"id": "6",
"description": "The hydraulic press achieves consistent force distribution across the entire surface of the workpiece.",
"Ollama_Response": "not problem"
},
{
"id": "7",
"description": "The electric motor operates with 95% energy efficiency, reducing operational costs significantly.",
"Ollama_Response": "not problem"

},
{
"id": "8",
"description": "The automated welding system ensures precision by adapting to minor variations in component alignment.",
"Ollama_Response": "not problem"
},
{
"id": "9",
"description": "The HVAC system maintains a stable temperature range, improving energy efficiency in extreme weather.",
"Ollama_Response": "not problem"
},
{
"id": "10",
"description": "The 3D printer produces high-resolution parts with complex geometries without requiring post-processing.",
"Ollama_Response": "not problem"
}
]