Extendable Tools (Orchestrator)

This guide outlines the standardized process for extending the AI Orchestrator system with custom tools. By following the steps below, developers can register new functions that the orchestrator will recognize and invoke automatically based on user input. The system is designed with extensibility in mind, requiring minimal configuration for new capabilities.

Step 1: Define the Function

Create a Python function that accepts a single argument (`user_input: str`) and returns a string response. This function encapsulates the logic you want the orchestrator to perform.

```
def calculate_profit(user_input: str) -> str:
    """Calculates profit based on cost and selling price in the input."""
    return "Profit is ₹30"
```

Step 2: Update the `system_prompt`

Update the orchestrator's `system_prompt` to include a brief description of your function. This informs the LLM of the tool's purpose and when to invoke it.

```
system_prompt = """
Functions:
- check_in_document: For company or document-related queries.
- check_in_db: For product rates or inventory information.
- suggest_clothing_combination: For outfit recommendations.
- get_current_date: Returns today's date.
- calculate_profit: For computing profit based on input.
- your_new_function_name: Describe the function's purpose here.
"""
```

Step 3: Register the Tool in `tool_labels`

Add your new tool to the `tool_labels` dictionary. The key is the function name, and the value is the user-facing label.

```
tool_labels = {
    "check_in_document": "Company Info",
    "check_in_db": "Product Rates",
    "suggest_clothing_combination": "Clothing Suggestions",
    "get_current_date": "Current Date",
    "calculate_profit": "Profit Calculation",
    "your_new_function_name": "Your Tool Label",
}
```

Step 4: You're Done!

Once the above steps are complete, the orchestrator will automatically detect and utilize your function when appropriate based on user intent. No further configuration is needed.

Example: Adding a Currency Converter Tool

Function Definition

```
def convert_currency(user_input: str) -> str:
    """Converts currency from INR to USD based on the user input."""
    return "₹100 = $1.20"
```

Add to `system_prompt`

```
- convert_currency: Converts between currencies such as INR, USD, and EUR.
```

Add to `tool_labels`

```
tool_labels = {
    ...
    "convert_currency": "Currency Converter",
}
```

Quick Reference Summary

Step	Action Description
1	Define a new function with `user_input` as the parameter.
2	Add a description of the function to the `system_prompt`.
3	Register the function in the `tool_labels` dictionary.
4	The orchestrator will now auto-discover and use your tool.

For advanced tools or integrations (e.g., APIs, databases), ensure the logic is secure and handles edge cases gracefully.