MCP servers & custom tool integration

Test one tool before adding the next

Session M3-03 | 3 hours

Agenda

Segment	Time
MCP request path and boundaries	8 min
Names, descriptions, and schemas	10 min
One tool from schema to response	15 min
Bad input and visible errors	10 min
Test output and review	9 min
Lab handoff	8 min

The MCP request path

```
Client
  ↓ tools/list
Tool catalog
  ↓ tools/call
Schema validation
  ↓
Handler
  ↓
Success or visible error
```

MCP standardizes discovery and calls. It does not approve the server or make its result correct.

Five boundaries to review

Boundary	Review question
Catalog	Which capabilities can the client discover?
Schema	Which arguments are accepted?
Handler	What work happens after a call?
Result	Can the caller distinguish failure from success?
Transport	Which process or endpoint carries the request?

Start with stdio and synthetic data. Keep the first review small.

Start with one narrow capability

```
{
  "name": "get_weather",
  "description": "Return deterministic synthetic current weather for one city.",
  "inputSchema": {
    "type": "object",
    "required": ["city"],
    "additionalProperties": false
  }
}
```

Clients use the tool name, description, and schema as part of the interface.

Start with one tool because every exposed capability adds risk and maintenance.

Trace the good request

```
{
  "name": "get_weather",
  "arguments": {
    "city": "Oslo"
  }
}
```

```
{
  "city": "Oslo",
  "temperature": 30,
  "unit": "celsius",
  "conditions": "rain"
}
```

Deterministic synthetic output makes exact tests possible.

Trace the bad request

```
{
  "name": "get_weather",
  "arguments": {
    "city": "   "
  }
}
```

```
{
  "error": "city must be a non-empty string"
}
```

The MCP result sets `isError: true`. An invalid request must not look like an empty success response.

Schema and handler checks work together

Schema	Handler
Helps clients form valid calls	Protects direct and protocol calls
Documents required fields	Applies business limits
Rejects extra properties	Returns a stable error shape

Do not assume every caller reaches the handler through the same client-side check.

What each check shows

Check	Claim
Unit test	Handler behavior matches expectations for tested inputs
`tools/list`	The expected catalog is exposed
`tools/call`	The stdio protocol path reaches the handler
Copilot call	One client allowed by repository policy can discover and choose the tool

Keep the deterministic tests even when the client integration works.

Lab handoff

Apply the same contract to three tools. Check discovery, valid input, invalid input, and visible errors through tests and the MCP Inspector.

Submit a three-tool server with recorded requests, responses, and errors.