Lecture - 10

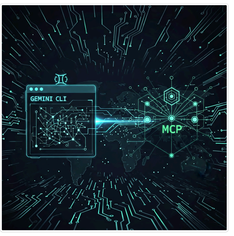
Gemini Code Assist Course

- → Gemini CLI + MCP Integration
- → MCP?
- → With MCP / Without MCP
- → Use cases in this lesson
- → Get Github PAT (Personal Access Token)
- → Configure MCP in Gemini CLI
- → Verify integration
- → Using the Github MCP Tool
- → Practice
- → Conclusion

Gemini CLI + MCP Integration →

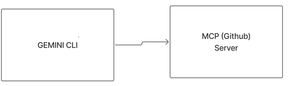
→ Until now, Gemini CLI worked with local files and commands

→ By integrating with Github through MCP, Gemini becomes more than a local assistant - CLOUD COLLABORATOR

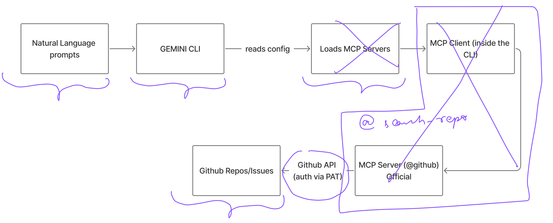


→ What is MCP? (Model Context Protocol)

- A standard format that lets AI Agents talk to external services clearly and consistently
- Predictable, structured data
- Efficient



→ With MCP / Without MCP



→ Use cases in this lesson -

- Configure Gemini CLI to connect with the official Github MCP Server
- ((Unlocks automation directly with Github repositories))

→ Get Github PAT (Personal Access Token)

Settings → Developer settings → Tokens (Classic)

→ Configure MCP in Gemini CLI →

- .gemini/settings.json

→ Verify the integration -

- Restart Gemini "Using 1 MCP Server"
- /mcp

→ Using the Github MCP Tool -

- MCP tool directly interacts with Github's API
- "@github search for repositories about Expense trackers"
- Gemini makes an authenticated req and returns structured results

→ Practice

→ Conclusion

- MCP extends Gemini beyond local development