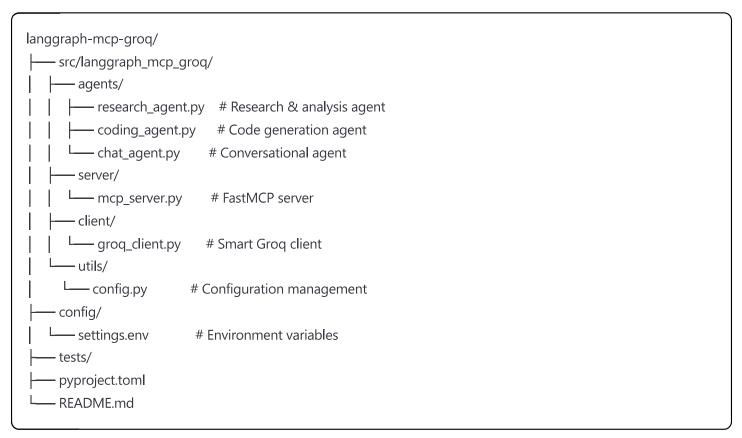
LangGraph MCP Groq Project

A complete implementation of LangGraph agents with FastMCP server integration and Groq LLM, featuring a smart client with automatic query routing.

Features

- LangGraph Agents: Research, Coding, and Chat agents with sophisticated workflows
- FastMCP Server: High-performance MCP server exposing agents as tools
- **Groq Integration**: Ultra-fast LLM inference with Groq
- Smart Client: Groq-powered query routing to the best agent
- Rich UI: Beautiful terminal interface with syntax highlighting

Project Structure



K Setup

1. Create Project Structure

bash

```
# Run the project setup script
uv init langgraph-mcp-groq
cd langgraph-mcp-groq

# Create directories
mkdir -p src/langgraph_mcp_groq/{agents,server,client,utils}
mkdir -p tests config scripts
```

2. Install Dependencies

bash

Install all dependencies

uv sync

3. Configure Environment

bash

Copy and edit settings

cp config/settings.env.example config/settings.env

Edit (config/settings.env):

env

GROQ_API_KEY=your_groq_api_key_here

MCP_SERVER_HOST=localhost

MCP_SERVER_PORT=8000

DEFAULT_MODEL=Ilama-3.1-70b-versatile

TEMPERATURE=0.7

MAX_TOKENS=4096

4. Get Groq API Key

- 1. Visit console.groq.com
- 2. Create account and generate API key
- 3. Add to config/settings.env



Start MCP Server

bash

Run the MCP server

uv run mcp-server

Or directly

uv run python -m langgraph_mcp_groq.server.mcp_server

Start Groq Client

bash

Run the interactive client

uv run groq-client

Or directly

uv run python -m langgraph_mcp_groq.client.groq_client

Client Modes

- 1. **Smart Mode**: Auto-routes queries to the best agent
- 2. **See Research Assistant**: Dedicated research and analysis
- 3. **Coding Assistant**: Code generation, testing, and review
- 4. **Chat Assistant**: General conversation with memory
- 5. **Server Status**: Check server and agent status
- 6. **Clear History**: Reset conversation history

Available Agents

Research Agent

- Comprehensive topic analysis
- Research summarization
- Source identification
- Key insights extraction

Coding Agent

- Multi-language code generation
- Automatic test creation

- Code explanation
- Code review and optimization

Chat Agent

- Natural conversation
- Context awareness
- Conversation summarization
- Multi-turn dialogue

N.

MCP Server Tools

research_assistant)

```
json
{
    "query": "Al trends in 2024"
}
```

Returns research results, summary, and sources.

coding_assistant

```
| json | {
| "task": "Create a REST API with FastAPI",
| "language": "Python" | }
```

Returns code, explanation, tests, and review.

chat_assistant

```
json
{
    "message": "Hello, how are you?",
    "session_id": "user123"
}
```

Returns conversational response with context.

agent_status

json

{}

Returns server status and agent information.



HTTP API Usage

Direct cURL Examples

```
bash
# Research Assistant
curl -X POST http://localhost:8000/tools/research_assistant \
 -H "Content-Type: application/json" \
 -d '{"query": "latest developments in quantum computing"}'
# Coding Assistant
curl -X POST http://localhost:8000/tools/coding_assistant \
 -H "Content-Type: application/json" \
 -d '{"task": "create a binary search function", "language": "Python"}'
# Chat Assistant
curl -X POST http://localhost:8000/tools/chat_assistant \
 -H "Content-Type: application/json" \
 -d '{"message": "Explain machine learning", "session_id": "demo"}'
# Server Status
curl -X POST http://localhost:8000/tools/agent_status \
 -H "Content-Type: application/json" \
 -d '{}'
```

Development

Run Tests

bash

uv run pytest

Code Formatting

bash

uv run black src/ tests/

uv run flake8 src/ tests/

Type Checking

bash

uv run mypy src/

Customization

Add New Agent

- 1. Create agent in (src/langgraph_mcp_groq/agents/)
- 2. Implement LangGraph workflow
- 3. Add tool to MCP server
- 4. Update client interface

Example New Agent:

Configure Models

Available Groq models:

- (Ilama-3.1-405b-reasoning) (Most capable)
- (Ilama-3.1-70b-versatile) (Balanced)
- (llama-3.1-8b-instant) (Fastest)
- (mixtral-8x7b-32768) (High context)

Troubleshooting

Common Issues

1. Groq API Key Error

```
# Check your API key
echo $GROQ_API_KEY
```

2. Server Connection Error

```
bash
# Check if server is running
curl http://localhost:8000/tools/agent_status
```

3. **Dependencies Error**

```
# Reinstall dependencies
uv sync --reinstall
```

Debug Mode

```
# Run with debug logging

PYTHONPATH=src uv run python -c "
import logging
logging.basicConfig(level=logging.DEBUG)
from langgraph_mcp_groq.server.mcp_server import main
main()
"
```

License

MIT License - see LICENSE file for details.

Contributing

- 1. Fork the repository
- 2. Create feature branch
- 3. Add tests for new features
- 4. Submit pull request

® Roadmap

- Add more specialized agents
- ☐ Implement agent chaining
- Add web interface
- $\hfill \square$ Support for other LLM providers
- Agent performance metrics
- $\hfill \square$ Kubernetes deployment configs