

Study Tools MCP

python 3.10+

FastAPI latest

MCP Server

license MIT

An AI-powered study assistant built with Model Context Protocol (MCP) that generates quizzes, flashcards, summaries, and concept explanations from your study materials.

💡 Features

- **Smart Summarization** - Generate concise summaries from study materials
- **Quiz Generation** - Create customizable quizzes with difficulty levels
- **Concept Explanation** - Get beginner/intermediate/advanced explanations
- **Flashcards** - Auto-generate flashcard decks from documents
- **Comparison Tool** - Compare and contrast multiple concepts
- **MCP Integration** - Works directly with Claude Desktop
- **Web UI** - Standalone chat interface with FastAPI backend

🛠 Tech Stack

- **Backend:** FastAPI + Python 3.10
- **MCP:** Model Context Protocol server/client
- **AI:** OpenAI API
- **Document Parsing:** PyPDF2, pdfplumber, python-docx
- **Frontend:** Vanilla JavaScript, HTML, CSS

🚀 Quick Start

Prerequisites

- Python 3.10+
- OpenAI API key

Installation

1. Clone the repository:

```
git clone <your-repo-url>
cd 4.study-tools-mcp
```

2. Install dependencies:

```
pip install -e .
```

3. Create `.env` file:

```
cp .env.example .env  
# Edit .env and add your OPENAI_API_KEY
```

4. Add your study materials:

Place PDF or Markdown files in `data/notes/`:

```
data/notes/  
└── Machine Learning.pdf  
└── Your Notes.md
```

5. Run the application:

```
python app.py
```

6. Open browser:

```
http://localhost:8080
```

Docker Deployment

Build and Run

```
docker build -t study-tools-mcp .  
docker run -p 8080:8080 --env-file .env -v ./data/notes:/app/data/notes study-tools-mcp
```

Project Structure

```
study-tools-mcp/  
└── app.py                                # FastAPI web application  
└── src/study_tools_mcp/  
    ├── server.py                            # MCP server entry point  
    └── config.py                           # Configuration
```

```

    └── tools/           # Quiz, flashcards, summarizer, explainer
    └── parsers/        # PDF and Markdown parsers
    └── utils/          # Logger
    └── static/          # Frontend assets
    └── templates/       # HTML templates
    └── data/notes/      # Your study materials
    └── logs/            # Application logs
    └── pyproject.toml   # Dependencies

```

💻 Usage

Web UI

The web interface provides an interactive chat where you can ask the AI to:

Tool	Example Prompt
Summarize	Summarize the topic: neural networks
Quiz	Create a 5-question quiz on "decision trees" at intermediate level
Explain	Explain the concept "gradient descent" at beginner level
Compare	Compare these concepts: SVM KNN
Flashcards	Create 10 flashcards for: ensemble methods

Claude Desktop Integration

Add to %APPDATA%\Claude\claude_desktop_config.json:

```
{
  "mcpServers": {
    "study-tools-mcp": {
      "command": "uv",
      "args": ["--directory", "C:\\path\\to\\study-tools-mcp", "run", "study-tools-mcp"]
    }
  }
}
```

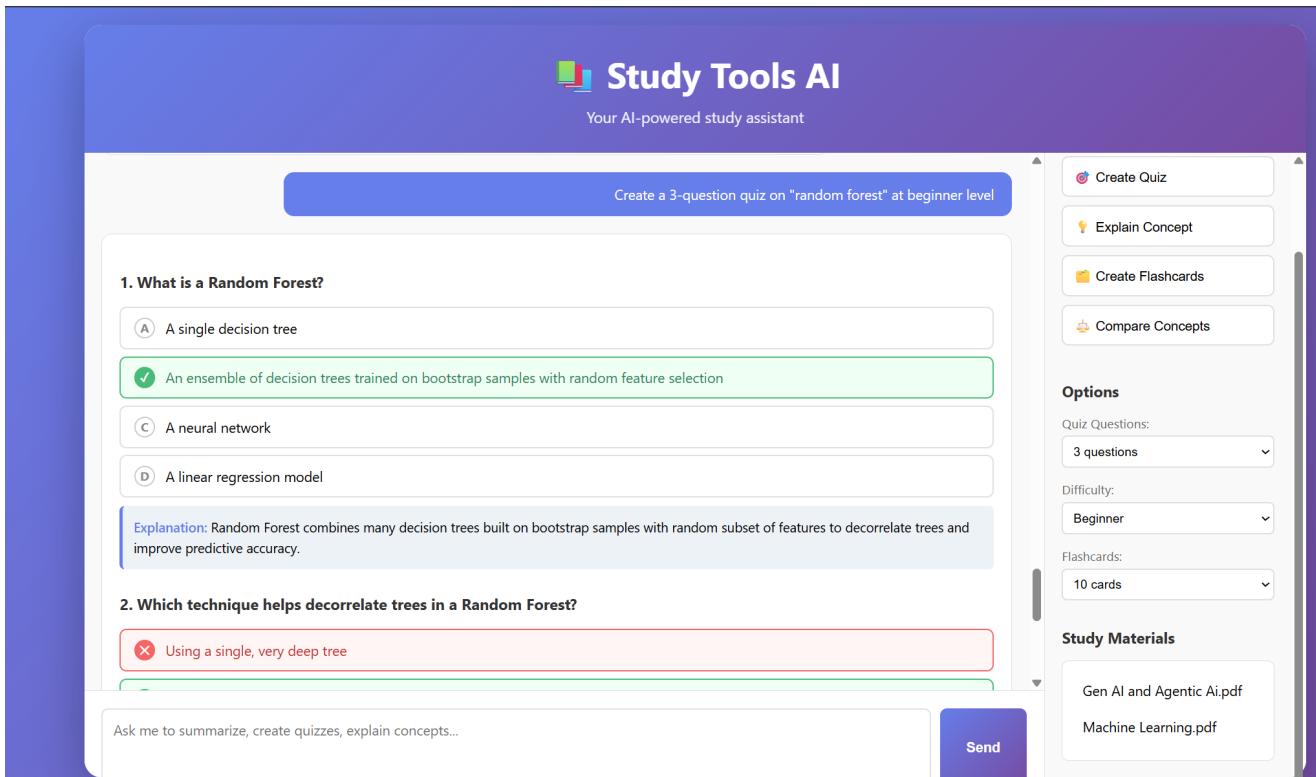
Restart Claude Desktop - the tools will be available automatically.

📡 API Endpoints

- `GET /` - Web UI
- `GET /health` - Health check
- `GET /api/files` - List available study materials
- `POST /api/chat` - Chat with streaming

- POST /api/chat/clear - Clear conversation history

Screenshots



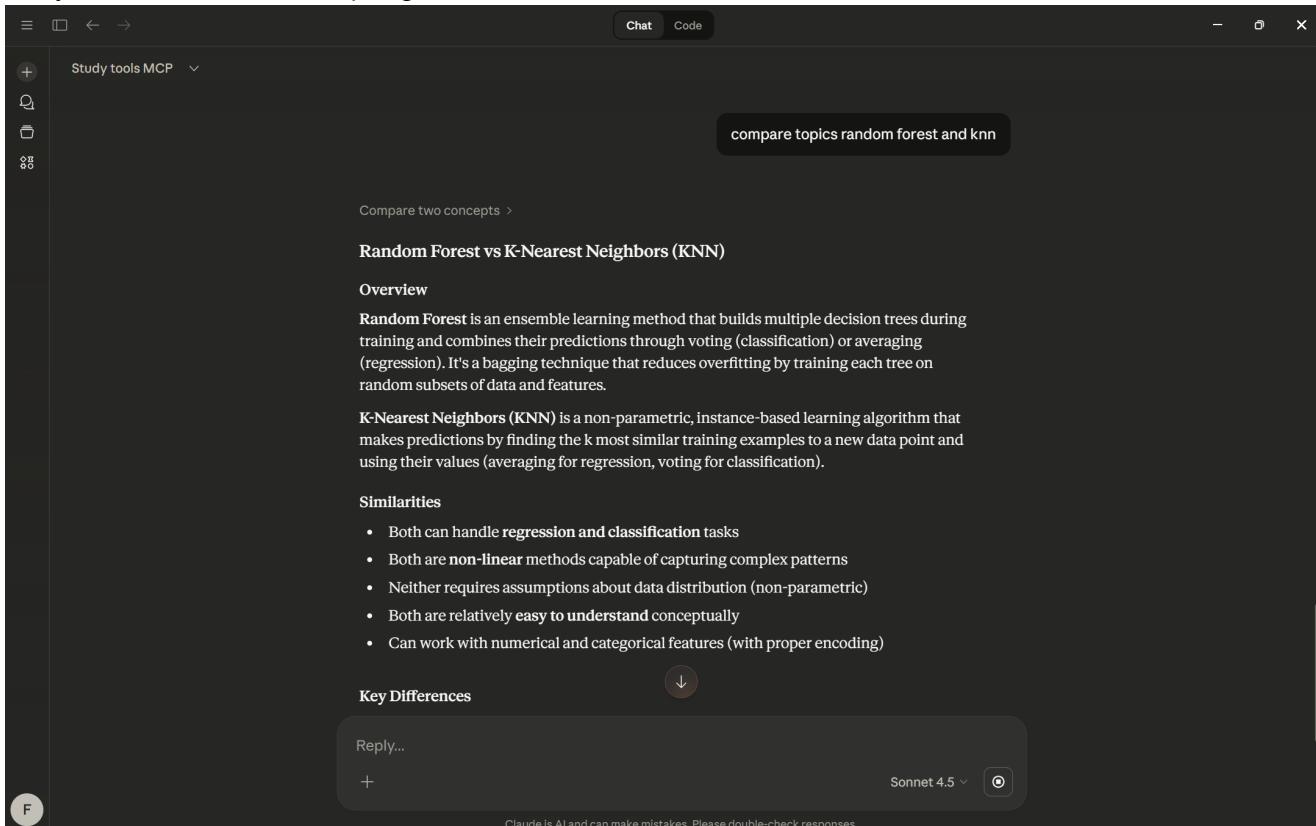
The screenshot shows the Study Tools AI interface. At the top, it says "Study Tools AI" and "Your AI-powered study assistant". A central box prompts "Create a 3-question quiz on 'random forest' at beginner level". Below this, there are two questions:

- 1. What is a Random Forest?**
 - (A) A single decision tree
 - (B) An ensemble of decision trees trained on bootstrap samples with random feature selection
 - (C) A neural network
 - (D) A linear regression model

Explanation: Random Forest combines many decision trees built on bootstrap samples with random subset of features to decorrelate trees and improve predictive accuracy.
- 2. Which technique helps decorrelate trees in a Random Forest?**
 - (X) Using a single, very deep tree

At the bottom, there's a text input field "Ask me to summarize, create quizzes, explain concepts..." and a blue "Send" button. To the right, there's a sidebar with options like "Create Quiz", "Explain Concept", "Create Flashcards", and "Compare Concepts". It also includes settings for "Quiz Questions: 3 questions", "Difficulty: Beginner", and "Flashcards: 10 cards". A "Study Materials" section lists "Gen AI and Agentic AI.pdf" and "Machine Learning.pdf".

Study Tool AI Interface with quiz generation



The screenshot shows the Study Tools MCP interface integrated with Claude code desktop. The top bar shows "Study tools MCP". The main content area displays a comparison between "Random Forest" and "K-Nearest Neighbors (KNN)".

Compare topics random forest and knn

Compare two concepts >

Random Forest vs K-Nearest Neighbors (KNN)

Overview

Random Forest is an ensemble learning method that builds multiple decision trees during training and combines their predictions through voting (classification) or averaging (regression). It's a bagging technique that reduces overfitting by training each tree on random subsets of data and features.

K-Nearest Neighbors (KNN) is a non-parametric, instance-based learning algorithm that makes predictions by finding the k most similar training examples to a new data point and using their values (averaging for regression, voting for classification).

Similarities

- Both can handle regression and classification tasks
- Both are non-linear methods capable of capturing complex patterns
- Neither requires assumptions about data distribution (non-parametric)
- Both are relatively easy to understand conceptually
- Can work with numerical and categorical features (with proper encoding)

Key Differences

Reply... Sonnet 4.5

Claude is AI and can make mistakes. Please double-check responses.

Study Tool AI Integration with Claude code desktop

License

