F-23 - MCP Servers in GitHub Copilot

SUMMARY

In this lesson, we explored the integration of MCP (Model Context Protocol) servers with GitHub Copilot, focusing on how they enhance productivity by automating GitHub and third-party app interactions directly from the IDE. We learned to use GitHub Copilot to automate tasks like creating repositories, pushing files, and managing pull requests, utilizing a personal access token for permissions.

Step-by-Step Guide

1. **Set Up MCP Server:** We installed an MCP server for GitHub using a package named `@model-context-protocol/server-GitHub`. This was done within a `.vscode` folder using a JSON configuration file to manage the server details and token.
2. **Generate Personal Access Token:** We generated a GitHub Personal Access Token with specific permissions (such as read/write access to repositories) to authorize our Copilot to interact with GitHub on our behalf.
3. **Create and Push Code:** We instructed GitHub Copilot to create a "Hello World" script, generate a new GitHub repository named "hello-world-repo", and push the code to this repository.
4. **Branch Management and PR Creation:** We further extended Copilot's capabilities by changing the script to "Hello Universe", creating a new branch, committing the changes, and opening a pull request.
5. **Integration with Other Services:** We installed and configured an MCP server for Slack, enabling GitHub Copilot to send and manage messages directly within Slack channels, demonstrating the versatility of MCP servers beyond GitHub alone.

WHAT WE LEARNED

- How to set up an MCP server for GitHub Copilot.
- Generating and configuring GitHub Personal Access Tokens for automated tasks.
- Automating repository creation, file pushing, and pull requests.
- Integration of third-party services like Slack using MCP servers.

HOW WE CAN APPLY IT

- **Automated DevOps:** Automate Git interactions like creating branches and handling merge requests without leaving the IDE.
- **Continuous Integration:** Utilize MCP servers for triggering CI/CD pipelines by integrating task management directly into the development environment.
- **Cross-Platform Integrations:** Connect and automate various services like Slack to streamline communication and workflows.

TIPS AND TRICKS

- **Secure Your Tokens:** Always store your GitHub Tokens securely. Consider using environment variables or secret managers in CI/CD.
- **Use Scope-Specific Tokens:** Generate tokens with only the necessary permissions for added security.
- **Explore Third-Party Integrations:** Leverage MCP servers to integrate tools like Trello, Sentry, and others directly into your workflow.

EXAMPLES

```
{
  "servers": {
    "github": {
      "api_token": "GITHUB_PAT_...",
      "args": "--foo bar"
    }
  }
}
```

Example Script Initialization

```
mkdir .vscode
touch .vscode/mcp.json
```

JavaScript: List Top 10 Prime Numbers

```
function isPrime(num) {
    for(let i = 2; i < num; i++)
        if(num % i === 0) return false;
    return num !== 1;
}
```

```
const primes = [];
let number = 2;

while (primes.length < 10) {
    if (isPrime(number)) primes.push(number);
    number++;
}

console.log(primes);
```

Slack Integration Example

```
{
  "servers": {
    "slack": {
      "bot_token": "YOUR_SLACK_BOT_TOKEN",
      "team_id": "YOUR_TEAM_ID"
    }
  }
}
```

With this setup, we demonstrated how GitHub Copilot's MCP servers can automate and streamline the development process directly from the code editor, enhancing both productivity and integration capabilities.