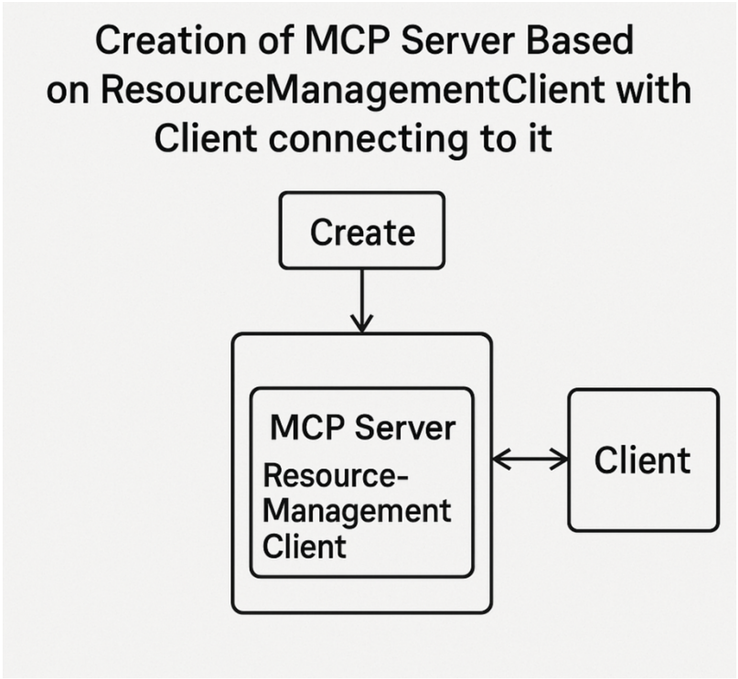
Azure Resource Management
MCP Server

Overview



this will be using the **Azure SDK for Python**,
and **ResourceManagementClient.resource_groups.create_or_update()** is a **Python SDK method** that internally calls the **Azure Resource Manager (ARM) REST API**.

✅ Summary

- **What you're using**:
  Python SDK (azure-mgmt-resource) for managing Azure resources.
- **API behind the scenes**:
  This SDK method wraps around the **Azure REST API** endpoint:

  bash
  PUT
  https://management.azure.com/subscriptions/{subscriptionId}/resourcegroups/{resourceGroupName}?api-version=2021-04-01

  This is part of the Azure Resource Manager REST API.

Key Features

1. **Resource Group Management**: Create and list Azure resource groups

Prerequisites

Before you begin, ensure you have the following installed and configured:

Required Software

- **Python 3.10 or higher**
- **Azure CLI** - For authentication (recommended approach)
- **uv package manager** - Faster alternative to pip for package management

Azure Requirements

- **Active Azure Subscription** with appropriate permissions

Authentication Setup (Recommended: Azure CLI)

The easiest way to authenticate is using Azure CLI:

1. Install Azure CLI from https://docs.microsoft.com/en-us/cli/azure/install- azure-cli
2. Login to your Azure account:

```
az login
```

3. Set your default subscription:

```
az account set --subscription "your-subscription-id"
```

Project Setup

Installing uv Package Manager

If you don't have uv installed yet:

```
pip install uv
```

Creating a Project Environment

Create a new project environment using uv:

```
uv init azure_rg_mcp_project
cd azure_rg_mcp_project
```

Adding Python Scripts

After setting up the project environment, you need to add the required Python scripts to your project directory:

1. **Place the server script**: Save the `azure_rg_server.py` file in your project directory
2. Your project structure should look like

```
azure_sql_mcp_project/
├── azure_rg_server.py
├── .env
└── pyproject.toml
```

Installing Required Packages

Install all required packages using uv:

```
uv add azure-core==1.34.0 azure-identity==1.23.0 azure-mgmt-resource==23.4.0
"mcp[cli]==1.9.2" python-dotenv==1.1.0
```

Environment Configuration

Create `.env` file in your project directory with the following configuration:

```
# Required: Your Azure subscription ID
AZURE_SUBSCRIPTION_ID=your_azure_subscription_id_here

# Optional: Service Principal authentication (if not using Azure CLI)
AZURE_TENANT_ID=your_tenant_id_here
AZURE_CLIENT_ID=your_client_id_here
AZURE_CLIENT_SECRET=your_client_secret_here
```

Note: If you're using Azure CLI authentication, you only need to set `AZURE_SUBSCRIPTION_ID` .

Running the Server

The system consists of two components that need to be run separately

```
# Run the server directly

Activate the environment
source .venv/bin/activate
```

```
python azure_rg_server.py
```

Debug Mode (Inspector)

For testing and debugging the server without

the client First run the server:

Then run the Inspector:

```
mcp dev azure_rg_server.py
```