MCP: Amazon S3 Demo Technical Requirements

This demonstration focuses on practical implementation aspects of MCP, with particular emphasis on AWS S3 integration. We'll walk through the essential components needed to get a functioning system up and running quickly.

AWS Components

- **AWS CLI installation** and configuration
- S3 bucket operations (list, upload, URL generation)
- Credential management and security practices

GitHub and Others

- Repository cloning and management
- Virtual environment setup with Python
- Dependency management using uv/python
- Code interaction via Cursor or similar tools

AWS CLI Installation and Setup

The installation process varies slightly between operating systems, but we'll focus on the macOS approach here. For Windows or Linux installations, refer to the official AWS documentation for platform-specific instructions.

Download the Installer Package

```
curl "https://awscli.amazonaws.com/AWSCLIV2.pkg" -o "AWSCLIV2.pkg"
```

This command fetches the official installer package from Amazon's servers, ensuring you're using the authentic software.

Install the AWS CLI

```
sudo installer -pkg AWSCLIV2.pkg -target /
```

Administrative privileges are required to install the software system-wide, making it available to all users on the machine.

Verify Installation

```
aws --version
```

This command should return the installed version number, confirming successful installation and readiness for configuration.

```
rahul@Mac ~ % aws --version
aws-cli/1.40.40 Python/3.12.
rahul@Mac ~ %
```

Configuring AWS Credentials

The AWS CLI provides a straightforward way to store your credentials locally, enabling authenticated access to your AWS resources without embedding sensitive information in your code.

These credentials should be treated with the utmost care, as they provide programmatic access to your AWS account and resources. Never share your credentials or commit them to version control systems.

Run Configuration Command

```
aws configure
```

This interactive command will prompt you for your credentials and default settings.

Enter Required Information

- AWS Access Key ID
- AWS Secret Access Key
- Default region (e.g., us-east-1)
- Output format (json recommended)

Verify Configuration

```
aws configure list
```

This command displays your current configuration without revealing sensitive key values.

For enhanced security, consider using IAM roles with temporary credentials for production environments, limiting the scope of permissions to only what's necessary for your specific use case.

Check AWS Config

.aws — -zsh — 80×24

```
rahul@Mac ~ % cd ~
rahul@Mac ~ % ls -la | grep .aws
drwxr-xr-x@   4 rahul  staff       128 Jun 20 23:34 .aws
drwxr-xr-x    4 rahul  staff       128 Jun 20 23:33 awscli-bundle
-rw-r--r--    1 rahul  staff  25533991 Jun 20 23:32 awscli-bundle.zip
rahul@Mac ~ %
rahul@Mac ~ %
rahul@Mac ~ % cd .aws
rahul@Mac .aws % ls
config          credentials
rahul@Mac .aws % cat config
[default]
region = us-east-2
output = json
rahul@Mac .aws %
rahul@Mac .aws % cat credentials
[default]
aws_access_key_id = [redacted]
aws_secret_access_key = [redacted]
rahul@Mac .aws %
```

Performing Basic AWS S3 Operations

Most frequently used S3 commands and operations that are likely incorporated into MCP workflows.

Common S3 Operations

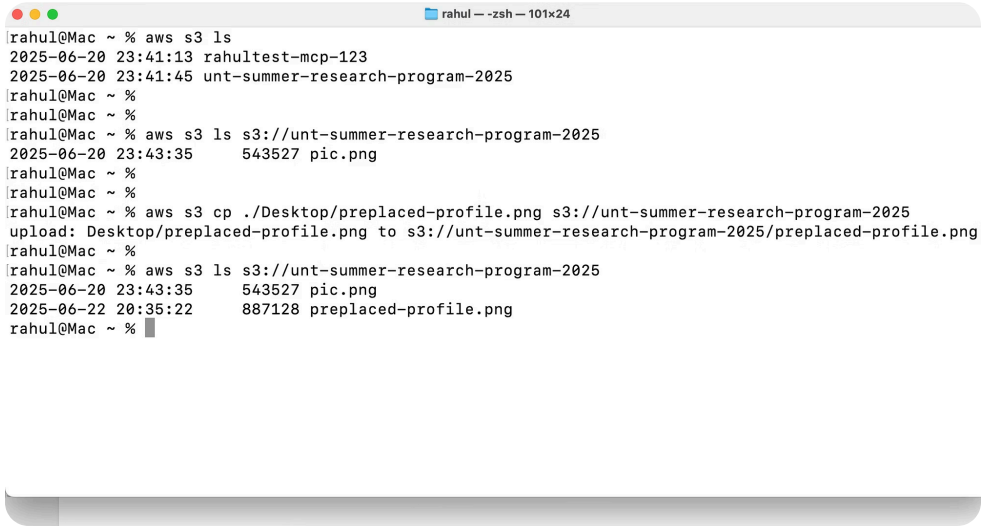
```
# List all S3 buckets
aws s3 ls

# List contents of a specific bucket
aws s3 ls s3://your-bucket-name/

# Upload a file to a bucket
aws s3 cp ./photo.jpg s3://your-bucket-name/

# Download a file from a bucket
aws s3 cp s3://your-bucket-name/photo.jpg ./

# Generate a pre-signed URL (temporary access)
aws s3 presign s3://your-bucket-name/photo.jpg
```



Setup MCP Server

Process of cloning a **repository** and setting up a proper development environment.

1 Clone the Repository

```
git clone https://github.com/aws-samples/sample-mcp-server-s3.git
```

This copies the remote repository to your local machine and moves you into the project directory, where all subsequent commands should be executed.

2 Create Virtual Environment

```
cd sample-mcp-server-s3

python3 -m venv .venv
source .venv/bin/activate
```

The virtual environment isolates your project dependencies from the system Python installation, preventing version conflicts between projects.

3 Sync Dependencies with uv

```
uv venv
uv sync
```

uv is a faster alternative to pip for managing Python dependencies, offering improved performance and consistency in dependency resolution.

Using Cursor to test MCP

Cursor is an AI-enhanced code editor that offers intelligent interaction with your repository, making it an excellent companion for MCP development. Installation

Download Cursor from the official website at [cursor.sh](). It's available for macOS, Windows, and Linux, with straightforward installation processes for each platform.

Opening Your Repository

Launch Cursor and select "Open Folder" to navigate to your cloned MCP repository. The editor will automatically detect project structure and language features.

Activating the Chat Panel

Access the Chat Panel through the sidebar or using keyboard shortcuts. This interface allows you to ask questions about your code or request modifications using natural language.