⭐ STEP 1 — Create GitHub Personal Access Token (PAT)

✓ Open link:

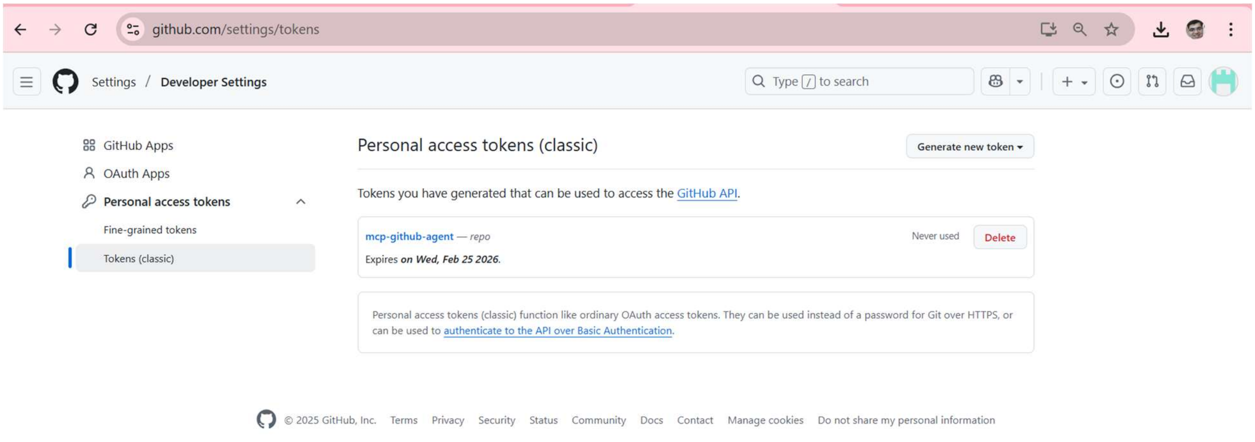
https://github.com/settings/personal-access-tokens/new

✓ Choose:

- **Token Type:** Fine-grained or Classic
- **Expiration:** 30 days
- **Scopes Required:**
    - `repo` (Read & Write)

✓ Copy your new PAT and save it temporarily.

📌 **Screenshot 1 Placeholder:** *(Insert screenshot of token generation page)*


github.com/settings/tokens
Settings / Developer Settings
Type / to search
GitHub Apps
OAuth Apps
Personal access tokens
Fine-grained tokens
Tokens (classic)
Generate new token ▾
Delete
© 2025 GitHub, Inc. Terms Privacy Security Status Community Docs Contact Manage cookies Do not share my personal information

Personal access tokens (classic)

Tokens you have generated that can be used to access the GitHub API.

mcp-github-agent — *repo* Never used

Expires *on Wed, Feb 25 2026.*

Personal access tokens (classic) function like ordinary OAuth access tokens. They can be used instead of a password for Git over HTTPS, or can be used to authenticate to the API over Basic Authentication.

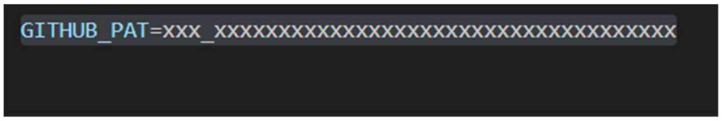
⭐ STEP 2 — Store Token Securely in `.env`

Create a file in your Task-6 folder:

.env

```
GITHUB_MCP_PAT=your_token_here
```

📌 **Screenshot 2 Placeholder:** *(Insert screenshot of .env file — blur token)*



```
GITHUB_PAT=xxx_xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx
```

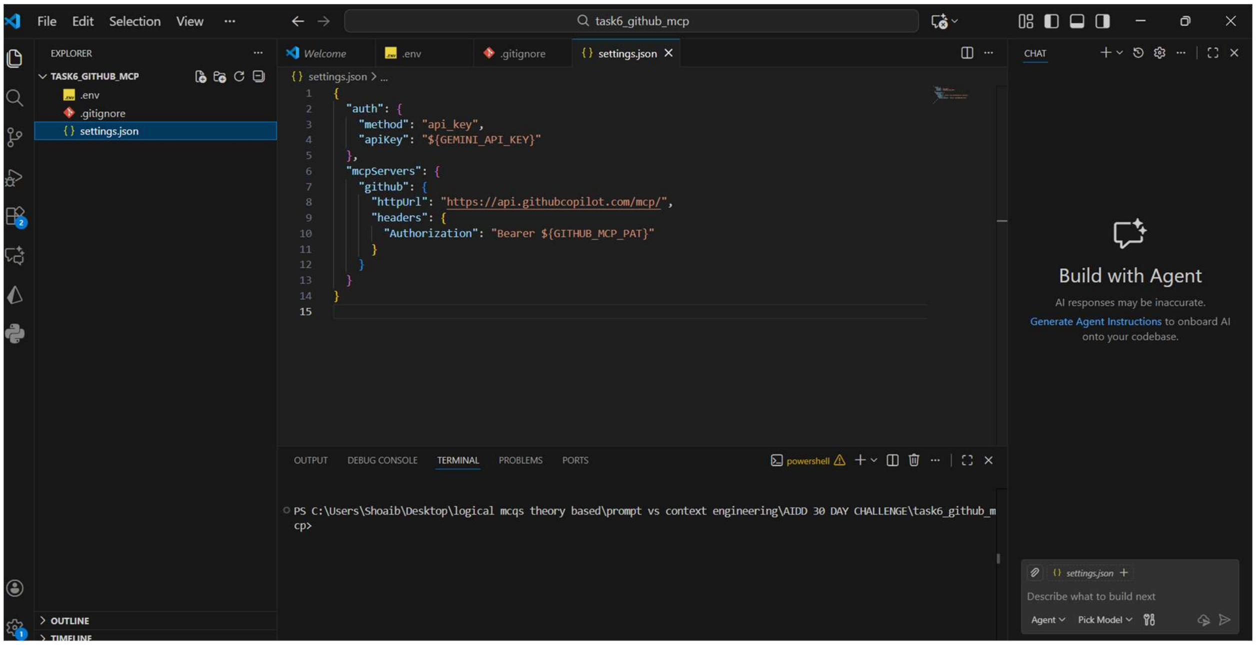
⭐ STEP 3 — Configure Gemini CLI to Use GitHub MCP Server

Open or create the file:

C:\Users<your-username>.gemini\settings.json

Paste the following configuration:

📌 **Screenshot 3 Placeholder:** *(Insert screenshot of the settings.json file)*



```json
{
  "auth": {
    "method": "api_key",
    "apiKey": "${GEMINI_API_KEY}"
  },
  "mcpServers": {
    "github": {
      "httpUrl": "https://api.githubcopilot.com/mcp/",
      "headers": {
        "Authorization": "Bearer ${GITHUB_MCP_PAT}"
      }
    }
  }
}
```

```
PS C:\Users\Shoaib\Desktop\logical mcqs theory based\prompt vs context engineering\AIDD 30 DAY CHALLENGE\task6_github_m
cp>
```

Build with Agent

AI responses may be inaccurate.

Generate Agent Instructions to onboard AI onto your codebase.

Describe what to build next

✓ Clean
✓ Official format
✓ Token auto-loads from `.env`
✓ Perfect for Hosted MCP

⭐ STEP 4 — Restart Gemini CLI

In terminal:

```
gemini logout
gemini login
```

If login cannot occur due to Gemini API key issues, you can still continue — Hosted MCP works without Gemini authentication.

⭐ STEP 5 — Verify MCP Connection

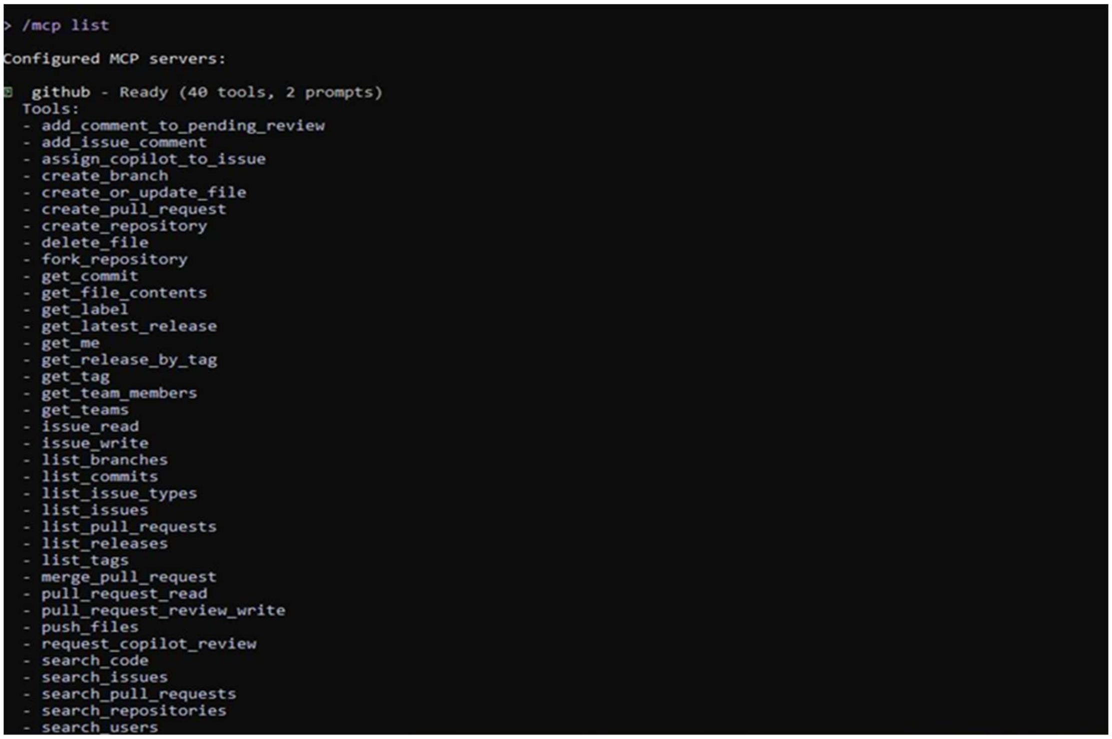
Run:

```
gemini mcp list
```

Expected Output:

```
github — Ready (40+ tools)
```

📌 **Screenshot 4 Placeholder:** *(Insert screenshot of 'gemini mcp list' showing GitHub enabled)*



```
> /mcp list

Configured MCP servers:

  github - Ready (40 tools, 2 prompts)
  Tools:
  - add_comment_to_pending_review
  - add_issue_comment
  - assign_copilot_to_issue
  - create_branch
  - create_or_update_file
  - create_pull_request
  - create_repository
  - delete_file
  - fork_repository
  - get_commit
  - get_file_contents
  - get_label
  - get_latest_release
  - get_me
  - get_release_by_tag
  - get_tag
  - get_team_members
  - get_teams
  - issue_read
  - issue_write
  - list_branches
  - list_commits
  - list_issue_types
  - list_issues
  - list_pull_requests
  - list_releases
  - list_tags
  - merge_pull_request
  - pull_request_read
  - pull_request_review_write
  - push_files
  - request_copilot_review
  - search_code
  - search_issues
  - search_pull_requests
  - search_repositories
  - search_users
```

⭐ STEP 6 — Test GitHub MCP Server

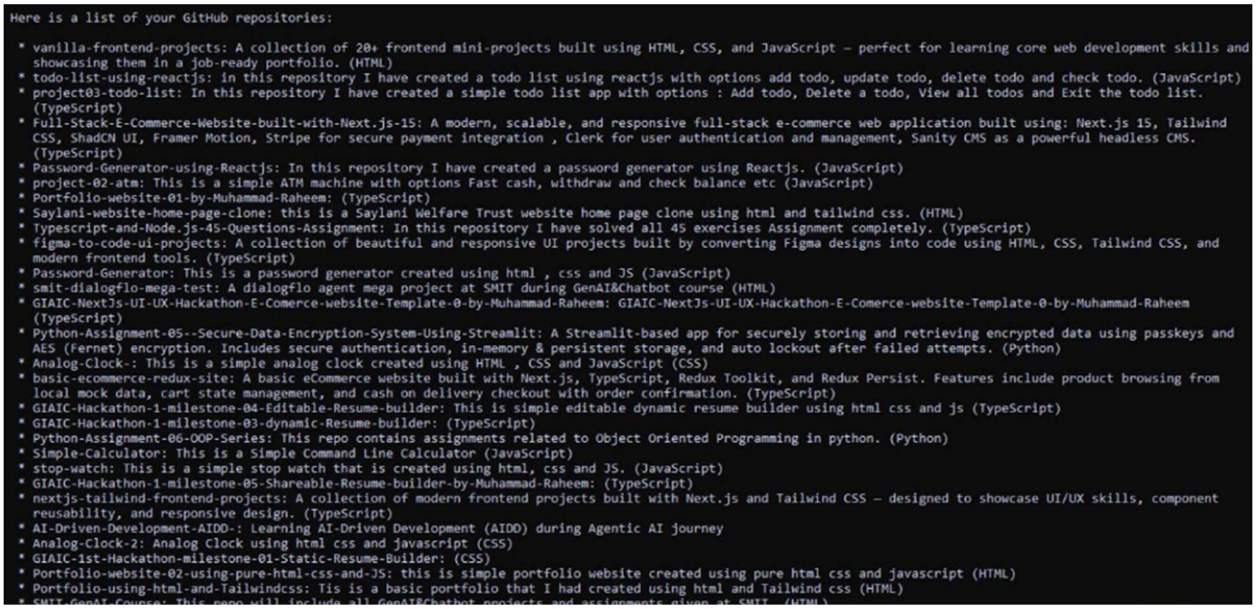
Ask Gemini:

```
List my GitHub repositories
```

If MCP is correctly connected, Gemini will respond with:

- Repo names
- Repo URLs
- Public/private details

📌 **Screenshot 5 Placeholder:** *(Insert screenshot of "List my GitHub repositories" output)*



```
Here is a list of your GitHub repositories:

 * vanilla-frontend-projects: A collection of 20+ frontend mini-projects built using HTML, CSS, and JavaScript – perfect for learning core web development skills and
   showcasing them in a job-ready portfolio. (HTML)
 * todo-list-using-reactjs: in this repository I have created a todo list using reactjs with options add todo, update todo, delete todo and check todo. (JavaScript)
 * project03-todo-list: In this repository I have created a simple todo list app with options : Add todo, Delete a todo, View all todos and Exit the todo list.
   (TypeScript)
 * Full-Stack-E-Commerce-Website-built-with-Next.js-15: A modern, scalable, and responsive full-stack e-commerce web application built using: Next.js 15, Tailwind
   CSS, ShadCN UI, Framer Motion, Stripe for secure payment integration , Clerk for user authentication and management, Sanity CMS as a powerful headless CMS.
   (TypeScript)
 * Password-Generator-using-Reactjs: In this repository I have created a password generator using Reactjs. (JavaScript)
 * project-02-atm: This is a simple ATM machine with options Fast cash, withdraw and check balance etc (JavaScript)
 * Portfolio-website-01-by-Muhammad-Raheem: (TypeScript)
 * Saylani-website-home-page-clone: this is a Saylani Welfare Trust website home page clone using html and tailwind css. (HTML)
 * Typescript-and-Node.js-45-Questions-Assignment: In this repository I have solved all 45 exercises Assignment completely. (TypeScript)
 * figma-to-code-ui-projects: A collection of beautiful and responsive UI projects built by converting Figma designs into code using HTML, CSS, Tailwind CSS, and
   modern frontend tools. (TypeScript)
 * Password-Generator: This is a password generator created using html , css and JS (JavaScript)
 * smit-dialogflo-mega-test: A dialogflo agent mega project at SMIT during GenAI&Chatbot course (HTML)
 * GIAIC-NextJs-UI-UX-Hackathon-E-Comerce-website-Template-0-by-Muhammad-Raheem: GIAIC-NextJs-UI-UX-Hackathon-E-Comerce-website-Template-0-by-Muhammad-Raheem
   (TypeScript)
 * Python-Assignment-05--Secure-Data-Encryption-System-Using-Streamlit: A Streamlit-based app for securely storing and retrieving encrypted data using passkeys and
   AES (Fernet) encryption. Includes secure authentication, in-memory & persistent storage, and auto lockout after failed attempts. (Python)
 * Analog-Clock-: This is a simple analog clock created using HTML , CSS and JavaScript (CSS)
 * basic-ecommerce-redux-site: A basic eCommerce website built with Next.js, TypeScript, Redux Toolkit, and Redux Persist. Features include product browsing from
   local mock data, cart state management, and cash on delivery checkout with order confirmation. (TypeScript)
 * GIAIC-Hackathon-1-milestone-04-Editable-Resume-builder: This is simple editable dynamic resume builder using html css and js (TypeScript)
 * GIAIC-Hackathon-1-milestone-03-dynamic-Resume-builder: (TypeScript)
 * Python-Assignment-06-OOP-Series: This repo contains assignments related to Object Oriented Programming in python. (Python)
 * Simple-Calculator: This is a Simple Command Line Calculator (JavaScript)
 * stop-watch: This is a simple stop watch that is created using html, css and JS. (JavaScript)
 * GIAIC-Hackathon-1-milestone-05-Shareable-Resume-builder-by-Muhammad-Raheem: (TypeScript)
 * nextjs-tailwind-frontend-projects: A collection of modern frontend projects built with Next.js and Tailwind CSS – designed to showcase UI/UX skills, component
   reusability, and responsive design. (TypeScript)
 * AI-Driven-Development-AIDD-: Learning AI-Driven Development (AIDD) during Agentic AI journey
 * Analog-Clock-2: Analog Clock using html css and javascript (CSS)
 * GIAIC-1st-Hackathon-milestone-01-Static-Resume-Builder: (CSS)
 * Portfolio-website-02-using-pure-html-css-and-JS: this is simple portfolio website created using pure html css and javascript (HTML)
 * Portfolio-using-html-and-Tailwindcss: Tis is a basic portfolio that I had created using html and Tailwind css (HTML)
 * SMIT-GenAI-Course: This repo will include all GenAI&Chatbot projects and assignments given at SMIT. (HTML)
```