Custom GitHub Copilot Agent Approval Request

Date: December 10, 2025
Requestor: [Your Name]
Department: [Your Department]
Purpose: Enable creation of security-focused custom agents for vulnerability remediation

---

Executive Summary

We request approval to create and deploy a custom GitHub Copilot agent in our development environment. This agent will automate detection and remediation of Snyk security vulnerabilities across our repositories, improving our security posture and reducing remediation time.

Prerequisites

Before creating a custom GitHub Copilot agent, ensure you have:

- **GitHub Copilot Pro, Business, or Enterprise plan** – Custom agents require an active Copilot subscription[1]
- **Repository write access** – You must be able to create and commit to the target repository[1]
- **Familiarity with agent syntax** – Basic understanding of YAML configuration and Markdown[1]
- **Target repository identified** – Know which repository will host the agent definition
- **Agent purpose defined** – Have a clear use case and workflow in mind (e.g., Snyk vulnerability fixing)[1]
- **Git and GitHub.com access** – Ability to navigate the GitHub UI and commit code
- **Security approval** – If in an enterprise, ensure your organization permits custom agents[1]

What is a Custom Copilot Agent?

A custom agent is a specialized automation tool that extends GitHub Copilot's built-in coding agent with specific instructions, tools, and workflows tailored to our organization's needs. It is defined in code (`.github/agents/`) and version-controlled like any other repository artifact[1].

Key characteristics:

- Operates only within authorized repositories with explicit user action
- Runs under the same permissions as the triggering user
- Cannot access CI secrets or files outside the repository scope
- All actions are auditable and visible in GitHub's activity logs

Proposed Agent: Snyk Vulnerability Fixer

Objective: Automate the process of identifying, analyzing, and proposing fixes for Snyk-reported security vulnerabilities.

Scope:

- Read repository code and Snyk reports
- Analyze vulnerability details (type, severity, remediation steps)
- Edit source files to apply fixes (e.g., dependency updates, code patches)
- Create pull requests for review before any changes are merged

Limitations:

- The agent cannot push directly to `main` or `master` branches; all changes go to `copilot/*` branches
- The agent cannot access GitHub tokens with elevated permissions
- The agent can only operate in repositories where it is configured
- Pull requests require human review and approval before merging

---

Security Controls & Mitigations

Data Protection

- **No training data leakage:** Code in custom agents is not used to train Copilot models[1]
- **No CI secrets access:** Agent cannot read or exfiltrate CI/CD secrets or environment variables
- **Scoped context:** Agent only sees files in the current repository
- **Session-based tokens:** Agent tokens are revoked after each session[1]

Access Control

- **Write access required:** Only team members with write access to a repository can trigger the agent[1]
- **PR-based workflow:** All edits are staged as pull requests requiring human review
- **Branch restrictions:** Agent can only commit to `copilot/` prefixed branches[1]
- **Audit logs:** All agent actions are logged and visible in GitHub's activity feed

Agent Governance

- **Version control:** Agent configuration files (`.github/agents/*.agent.md`) are tracked in Git
- **Code review:** Changes to agent behavior go through standard pull request review
- **Organization-wide policies:** Enterprise can enforce agent policies via ruleset configuration[1]
- **User authorization:** Agent only acts on issues/PRs assigned or triggered by authorized users

---

Transparency & Accountability

- **No invisible directives:** All agent instructions are stored in public or controlled repository files
- **Prompt visibility:** Security team can review the exact prompt/instructions driving the agent
- **Audit trail:** GitHub audit logs track which user triggered the agent and what changes it made
- **No external calls:** Agent does not have internet access to exfiltrate data to external systems

---

How to Create a Custom Agent

Step 1: Navigate to GitHub Copilot Agents

1. Go to `https://github.com/copilot/agents` in your browser[1]
2. Sign in with your GitHub account
3. You will see the agent creation interface with a prompt box at the top

Step 2: Select Repository and Branch

1. In the agent creation panel, use the dropdown to select the **repository** where you want the agent to live[1]
2. Select the **branch** (typically `main` or `master`)[1]
3. Ensure you have write access to the repository you select

Step 3: Create the Agent File

1. Click the + button or **Create agent** option[1]
2. GitHub will create a new file named `.github/agents/<agent-name>.agent.md`[1]
3. The file opens with a template containing:
   - YAML front-matter (metadata section)
   - Markdown body (instructions to the agent)[1]

Step 4: Configure Agent Metadata

Edit the YAML front-matter at the top of the file:

name: Snyk Vulnerability Fixer
description: Analyzes Snyk reports and proposes security fixes
tools:

- read # Read repository files
- search # Search code
- edit # Edit files
- pull_requests # Create pull requests

Key fields:[1]

- `name` – Display name for the agent (visible in GitHub.com dropdown)
- `description` – What the agent does (shown in agent selection UI)
- `tools` – Array of capabilities (read, edit, search, pull_requests, etc.)[1]

- `target` – (Optional) Restrict to specific environment (vs-code, github-com, both)[1]

Step 5: Write Agent Instructions

Below the YAML, write clear instructions for the agent in Markdown:

Snyk Vulnerability Fixer

You are an AI agent that automatically fixes security vulnerabilities identified by Snyk.

Workflow

1. Read the latest Snyk report or vulnerability details from the issue/PR
2. Analyze the vulnerability type (dependency update, code patch, etc.)
3. Identify affected files and remediation steps
4. Edit files with the fix (update package versions, patch code, etc.)
5. Create a pull request with:
   - Clear title: "Fix [Vulnerability Name] reported by Snyk"
   - Detailed description of the vulnerability and fix
   - Link to Snyk report if available

Guidelines

- Always create a new branch prefixed with `copilot/`
- Do not push directly to main; changes must go through a PR
- Ensure fixes do not introduce breaking changes
- Test where possible within the agent's capabilities

Step 6: Commit the Agent

1. Scroll to the bottom of the file
2. Click **Commit** (or **Propose changes** if in a PR workflow)[1]
3. Add a commit message, e.g., "Create Snyk vulnerability fixer custom agent"
4. Select **Commit directly to main branch** (or create a PR if your branch is protected)[1]
5. Once merged, the agent is active and can be used in that repository[1]

How to Use a Custom Agent

Using the Agent in GitHub.com

1. **Navigate to your repository** on GitHub.com[1]
2. **Open an issue or pull request** related to the vulnerability you want to fix[1]
3. **Click the Copilot icon** or open the **Copilot coding agent** sidebar (usually top-right of the page)[1]
4. **Select your custom agent** from the agents dropdown (you will see "Snyk Vulnerability Fixer" or the name you set)[1]
5. **Type your task prompt**, for example:
   ```
   Analyze the Snyk vulnerabilities in this repo and propose fixes for the
   critical dependency issues in package.json
   ```

6. **Press Enter or click Run**[1]
7. The agent will:
    - Read the repository code and Snyk report
    - Edit files as needed
    - Create a pull request with proposed changes[1]
8. **Review the PR** – Check the changes, comment if needed, and merge if approved[1]

Using the Agent in VS Code (Local)

1. **In VS Code**, open the repository locally
2. Open the **GitHub Copilot Chat** panel (Ctrl+Shift+I or Cmd+Shift+I)[1]
3. In the agent selector dropdown, choose your **custom agent** (e.g., "Snyk Vulnerability Fixer")[1]
4. **Type your task prompt** in the chat, e.g.:
   Fix all critical Snyk vulnerabilities in this repo
5. The agent will:
    - Analyze your local code
    - Suggest or apply edits
    - Create branches and PRs[1]
6. **Review and push** the created PR to GitHub[1]

Running the Agent from an Issue

1. **Create or open a GitHub issue** describing the vulnerability[1]
2. In the issue, click **Copilot coding agent**[1]
3. Select your **custom agent** from the dropdown[1]
4. Type a prompt, e.g., "Fix the vulnerability described in this issue"[1]
5. The agent will create a PR addressing the issue[1]

Monitoring Agent Activity

- All agent-created PRs appear in your repository's **Pull Requests** tab
- Check GitHub's **Activity** log to see agent actions and audit trail[1]
- Review agent-generated commits in the **Commits** tab[1]

Implementation Plan

Phase 1: Prerequisites & Setup (Week 1)

- Verify your Copilot plan (Pro, Business, or Enterprise)
- Ensure repository write access
- Get security approval if required

Phase 2: Agent Creation (Week 2)

- Create `.github/agents/snyk-fixer.agent.md` in target repositories
- Define agent configuration (name, description, tools)
- Embed Snyk remediation workflow as instructions
- Commit and merge agent definition

Phase 3: Testing & Validation (Week 3)

- Test agent on non-critical repository with sample issues
- Verify edits and PR creation work as expected
- Validate that agent respects branch protection rules
- Document any adjustments needed

Phase 4: Deployment & Monitoring (Week 4+)

- Expand to additional repositories as approved
- Monitor agent activity and PR quality
- Gather developer feedback
- Adjust agent instructions if needed
- Monthly review of agent-generated PRs

Phase 5: Governance (Ongoing)

- Quarterly audit of agent configuration and permissions
- Update agent instructions based on security requirements
- Scale agent to additional use cases (static analysis, code formatting, etc.)

Risk Assessment

Risk	Likelihood	Severity	Mitigation
Unintended code changes	Low	Medium	PR review gate + testing in non-critical repos first
Prompt injection attacks	Low	Medium	All agent instructions in version-controlled files (no hidden inputs)
Over-privileged edits	Low	High	Branch restrictions + branch protection rules for `main`/`master`
Agent misuse	Very Low	Medium	Write access check + audit logging

---

Benefits

- **Faster vulnerability resolution:** Automate routine remediation steps
- **Consistency:** Apply standardized fix patterns across all repositories
- **Developer focus:** Free engineers to focus on complex vulnerabilities
- **Compliance:** Demonstrate proactive security posture and rapid remediation
- **Auditability:** All agent actions logged and traceable

Security Team Review Checklist

- [ ] Agent instructions reviewed and approved
- [ ] No sensitive data (API keys, tokens) embedded in agent config
- [ ] Branch protection rules configured for `main`/`master`
- [ ] Audit logging enabled
- [ ] Initial pilot repository identified and approved
- [ ] Escalation process defined (who to contact if agent behaves unexpectedly)

Questions & Contact

Security Review Contact: [Security Team Lead Name] – [email]
Agent Configuration Owner: [Your Name] – [Your Email]

For technical questions about custom agents, see the official GitHub documentation on preparing custom agents in an enterprise[1].

References

[1] GitHub. (2024). "Preparing to use custom agents in your enterprise." GitHub Docs. https://docs.github.com/en/copilot/how-tos/administer-copilot/manage-for-enterprise/manage-agents/prepare-for-custom-agents

[2] GitHub. (2025). "How GitHub's agentic security principles make our AI agents as secure as possible." GitHub Blog. https://github.blog/ai-and-ml/github-copilot/how-githubs-agentic-security-principles-make-our-ai-agents-as-secure-as-possible/

[3] GitHub. (2024). "Creating custom agents." GitHub Docs. https://docs.github.com/en/copilot/how-tos/use-copilot-agents/coding-agent/create-custom-agents