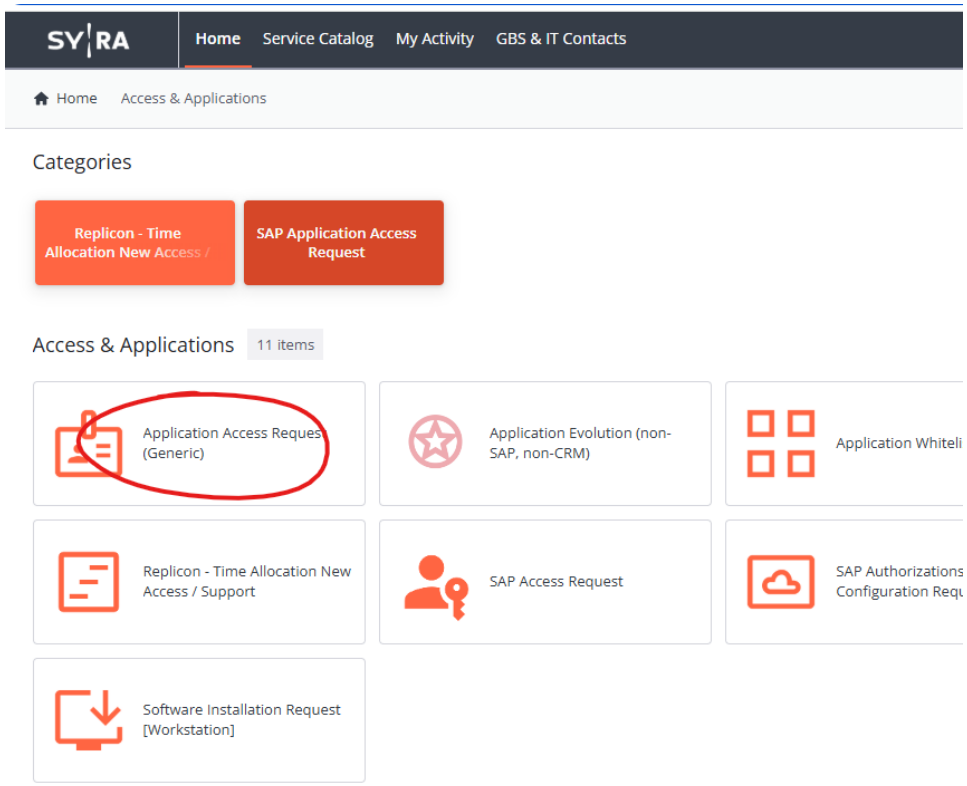
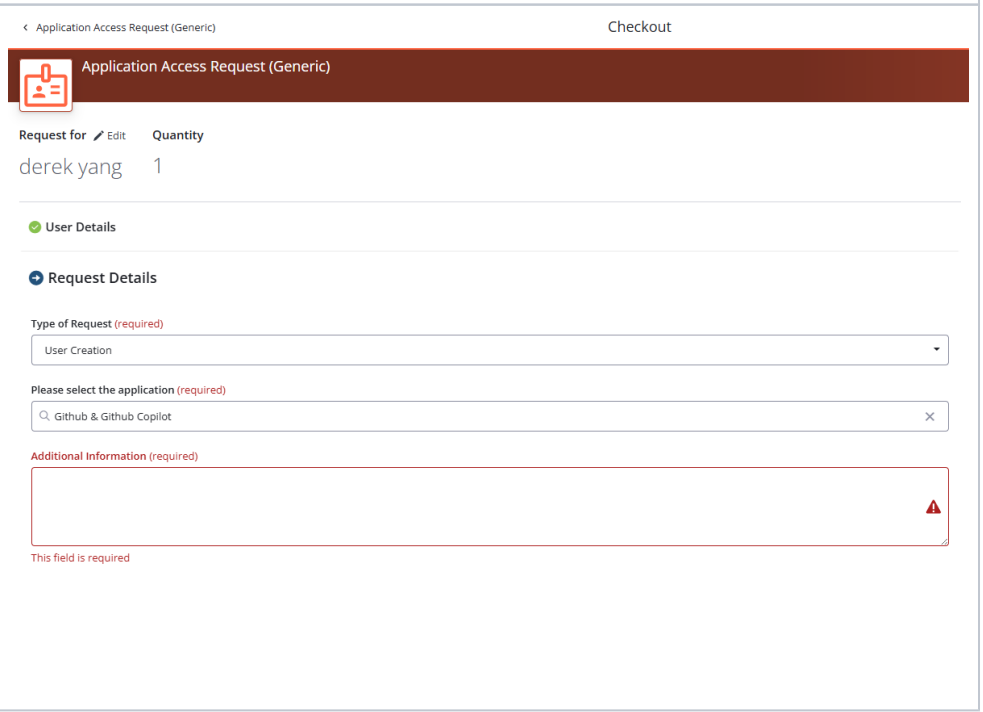
1. Onboarding & Governance

How to request Github / Copilot access

- Raise SYRA ticket from : https://syensqo-dwp.onbmc.com/dwp/app/#/page/w9ph6wjq/category/1039/SBE/SBE



- User creation request for GitHub & GitHub Copilot



Request format

User email list

Organization you have access approval (approval email from existing organization owner)

New organization request with justification

License required (Github or Github copilot)

Organization Naming Conventions

In the GitHub hierarchy, an **Organization** is the shared workspace that sits above **Teams** and acts as the owner of your **Repositories**.

If a Team is a "group of people," an Organization is the "company building" that houses the people, the projects, and the security front door.

1. Centralized Asset Ownership
2. The "Front Door" (Access Management)
3. Shared Resources & Policy Enforcement

- **Naming Convention (Org):** `SQO-<business-unit>` (e.g., `SQO-INFRASTRUCTURE`, `SQO-INTEGRATION` , `SQO-SYENSQOAI`).

- **Naming Convention (Repo):** Syensqo-[platform]-[app name/project name]
  - *platform:* (Azure), (GCP), (AWS), (Multi-cloud), (on-prem).
  - *Example:* `Syensqo-Azure-infra-Subscription-Mgmt`

Why do you need organization

We avoid "Org Sprawl." New Organizations are only created for distinct Business Units or large-scale Projects.

Policy: Strategic Organization Provisioning

To maintain a lean, secure, and manageable environment, we strictly adhere to a **"Zero Sprawl"** policy. New Organizations are not granted by default;

they are reserved exclusively for **distinct Business Units** with independent P&Ls or **large-scale, cross-functional projects** that require isolated governance.

Before a new Organization is provisioned, the requesting party must demonstrate that existing structures cannot meet their technical or regulatory requirements.

New Organization Request Form

Please provide the following information to the Architecture and Governance committee for review.

1. Existing Infrastructure Audit

- **Current Footprint:** Does your department or parent Business Unit already have an existing Organization?
- **Integration Analysis:** If yes, why can your requirements not be met by creating a new **Project** within the existing hierarchy?

2. Team Composition & Scale

- **Headcount:** What is the estimated number of internal users, external contractors, and administrators?

3. Strategic Justification

- **Business Necessity:** Explain the unique business driver for this separation (e.g., a legal divestiture, a joint venture, or a distinct brand identity).
- **Security & Compliance:** Are there specific regulatory requirements (e.g., HIPAA, PCI-DSS, or data residency laws) that mandate a "Hard Isolation" from the rest of the company's infrastructure?

4. Workload Profile & Roadmap

- **Current Scope:** Describe the primary applications, services, or workloads to be hosted immediately.
- **Utilization Forecast:** Provide a 12-to-24-month roadmap. What are the upcoming projects, and what is the expected growth in resource consumption (e.g estimation of license required, CI/CD pipeline usage?
- **Lifecycle:** Is this a permanent Business Unit or a time-bound project? (If time-bound, provide an estimated decommissioning date).

Team & Permission Management

In GitHub Enterprise Cloud,

a **Team** is much more than just a list of people. It is a fundamental organizational tool used to manage permissions, streamline communication, and group members based on their real-world roles or projects.

1. Access Control at Scale
2. Nested Teams (Hierarchy)
3. Mentioning and Communication
4. Required Reviews

- **Team Structure:** Always nest teams under the Parent Org.

- **Permissions:**
  - **Maintainer:** For dev /ops lead (manage settings, no billing)
  - **Write:** For Developers (standard CI/CD access).
  - **Triage/Read:** For Stakeholders and Auditors.

> **Info**
>
> **Rule:** Never grant individual permissions