4. Security and compliance
└─ **4.1 Authentication and authorization**
└─ **4.1.4 Permissions/Roles in GitHub**

---

1. What are the main types of roles in a GitHub repository?
2. What permissions does each role have in GitHub?
3. How do you assign and manage roles in a GitHub repository?
4. What is the difference between a repository collaborator and a team member?
5. How do you restrict branch access and actions in GitHub?
6. What is the purpose of code owners and how do they affect permissions?
7. How do organization-level permissions differ from repository-level permissions?
8. What are outside collaborators, and how are they managed?
9. What best practices should be followed for managing GitHub permissions?
10. How do you audit and review repository access in GitHub?

---

1. What are the main types of roles in a GitHub repository?
Owner, Admin, Maintainer, Write (Contributor), Triage, Read, and Outside Collaborator.

---

2. What permissions does each role have in GitHub?

- **Owner:** Full control over organization and all repositories.
- **Admin:** Full control over repository settings and content.
- **Maintainer:** Manages teams, can triage issues/PRs.
- **Write:** Push code, create branches, manage issues/PRs.
- **Triage:** Manage issues/PRs (label, assign), but cannot write code.
- **Read:** View content, clone repo, open issues/PRs.
- **Outside Collaborator:** Limited to explicit permissions given.

---

3. How do you assign and manage roles in a GitHub repository?
Go to *Repository → Settings → Manage Access*. Add users/teams and assign appropriate role.

---

4. What is the difference between a repository collaborator and a team member?

- A collaborator is granted access directly to a repository, typically outside the organization.
- A team member is part of an organization team, with permissions managed at the team level.

---

5. How do you restrict branch access and actions in GitHub?
Configure branch protection rules in repository settings:

- restrict pushes,
- require PR reviews,
- enforce status checks,
- and limit who can merge.

---

6. What is the purpose of code owners and how do they affect permissions?
Code owners are responsible for specific files or directories. Their review is required before changes to owned code can be merged, enforcing accountability.

7. How do organization-level permissions differ from repository-level permissions?

- *Organization-level permissions* apply across all repos and include actions like inviting members and managing billing.
- *Repository-level permissions* control access to specific repos only.

---

8. What are outside collaborators, and how are they managed?
Outside collaborators are users not part of the org but given access to specific repos. Managed by adding/removing their access in repo settings.

---

9. What best practices should be followed for managing GitHub permissions?

- Use least privilege principle.
- Prefer team-based over individual permissions.
- Regularly review and audit access.
- Use branch protections and required reviews.

---

11. **How do you audit and review repository access in GitHub?**

- Use the "Manage Access" tab to review users and roles.
- For organizations, use GitHub's audit log and periodic access reviews.