**Plan Agile with GitHub Projects and Azure Boards**

Advanced

Administrator

Developer

DevOps Engineer

Security Engineer

Security Operations Analyst

Service Adoption Specialist

Solution Architect

Technology Manager

Azure

Azure Artifacts

Azure Boards

Azure Cloud Services

Azure DevOps

Azure Pipelines

Azure Repos

Azure Test Plans

GitHub

This module introduces you to GitHub Projects, GitHub Project Boards and Azure Boards. It explores ways to link Azure Boards and GitHub, configure GitHub Projects and Project views, and manage work with GitHub Projects.

**Learning objectives**

By the end of this module, you're able to:

* Describe GitHub Projects and Azure Boards
* Link Azure Boards and GitHub
* Configure and Manage GitHub Projects and boards
* Customize Project views

[**Start**](https://learn.microsoft.com/en-us/training/modules/plan-agile-github-projects-azure-boards/1-introduction/)Add

**Prerequisites**

None

**This module is part of these learning paths**

* [AZ-400: Development for enterprise DevOps](https://learn.microsoft.com/training/paths/az-400-work-git-for-enterprise-devops/)

**Module assessment**

Assess your understanding of this module. Sign in and answer all questions correctly to earn a pass designation on your profile.

[**Take the module assessment**](https://learn.microsoft.com/training/modules/plan-agile-github-projects-azure-boards/13-knowledge-check/)

* [Introduction](https://learn.microsoft.com/en-us/training/modules/plan-agile-github-projects-azure-boards/1-introduction)3 min
* [Introduction to GitHub Projects and Project boards](https://learn.microsoft.com/en-us/training/modules/plan-agile-github-projects-azure-boards/2-introduction-to-project-boards)3 min
* [Introduction to Azure Boards](https://learn.microsoft.com/en-us/training/modules/plan-agile-github-projects-azure-boards/3-introduction-to-azure-boards)3 min
* [Configure projects and teams in Azure DevOps](https://learn.microsoft.com/en-us/training/modules/plan-agile-github-projects-azure-boards/4-configure-projects-teams-azure-devops)3 min
* [Link GitHub to Azure Boards](https://learn.microsoft.com/en-us/training/modules/plan-agile-github-projects-azure-boards/5-link-github-to-azure-boards)2 min
* [Configure GitHub Projects](https://learn.microsoft.com/en-us/training/modules/plan-agile-github-projects-azure-boards/6-configure)3 min
* [Manage work with GitHub Project boards](https://learn.microsoft.com/en-us/training/modules/plan-agile-github-projects-azure-boards/7-manage-work-github-project-boards)2 min
* [Customize Project views](https://learn.microsoft.com/en-us/training/modules/plan-agile-github-projects-azure-boards/8-customize-project-views)3 min
* [Collaborate using team discussions](https://learn.microsoft.com/en-us/training/modules/plan-agile-github-projects-azure-boards/9-collaborate-using-team-discussions)3 min
* [Design and implement a strategy for feedback cycles](https://learn.microsoft.com/en-us/training/modules/plan-agile-github-projects-azure-boards/10-design-implement-strategy-feedback-cycles)3 min
* [Design and implement source, bug, and quality traceability](https://learn.microsoft.com/en-us/training/modules/plan-agile-github-projects-azure-boards/11-design-implement-source-bug-quality-traceability)3 min
* [Agile Plan and Portfolio Management with Azure Boards](https://learn.microsoft.com/en-us/training/modules/plan-agile-github-projects-azure-boards/12-agile-plan-portfolio-management-azure-boards)57 min
* [Knowledge check](https://learn.microsoft.com/en-us/training/modules/plan-agile-github-projects-azure-boards/13-knowledge-check)3 min
* [Summary](https://learn.microsoft.com/en-us/training/modules/plan-agile-github-projects-azure-boards/14-summary)3 min

**Introduction**

100 XP

* 3 minutes

This module introduces you to GitHub Projects, GitHub Project Boards, and Azure Boards. It explores ways to link Azure Boards and GitHub, configure GitHub Projects and Project views, and manage work with GitHub Projects.

**Learning objectives**

After completing this module, students and professionals can:

* Describe GitHub Projects and Azure Boards.
* Link Azure Boards and GitHub.
* Configure and Manage GitHub Projects and boards.
* Customize Project views.

**Prerequisites**

* Understanding of what DevOps is and its concepts.
* Familiarity with version control principles is helpful but isn't necessary.
* Beneficial to have experience in an organization that delivers software.
* You need to create a GitHub account at GitHub.com and a project for some exercises. If you don't have it yet, see: [Join GitHub · GitHub](https://github.com/signup). If you already have your GitHub account, create a new repository [Creating a new repository - GitHub Docs](https://docs.github.com/repositories/creating-and-managing-repositories/creating-a-new-repository).

**Next unit: Introduction to GitHub Projects and Project boards**

**Introduction to GitHub Projects and Project boards**

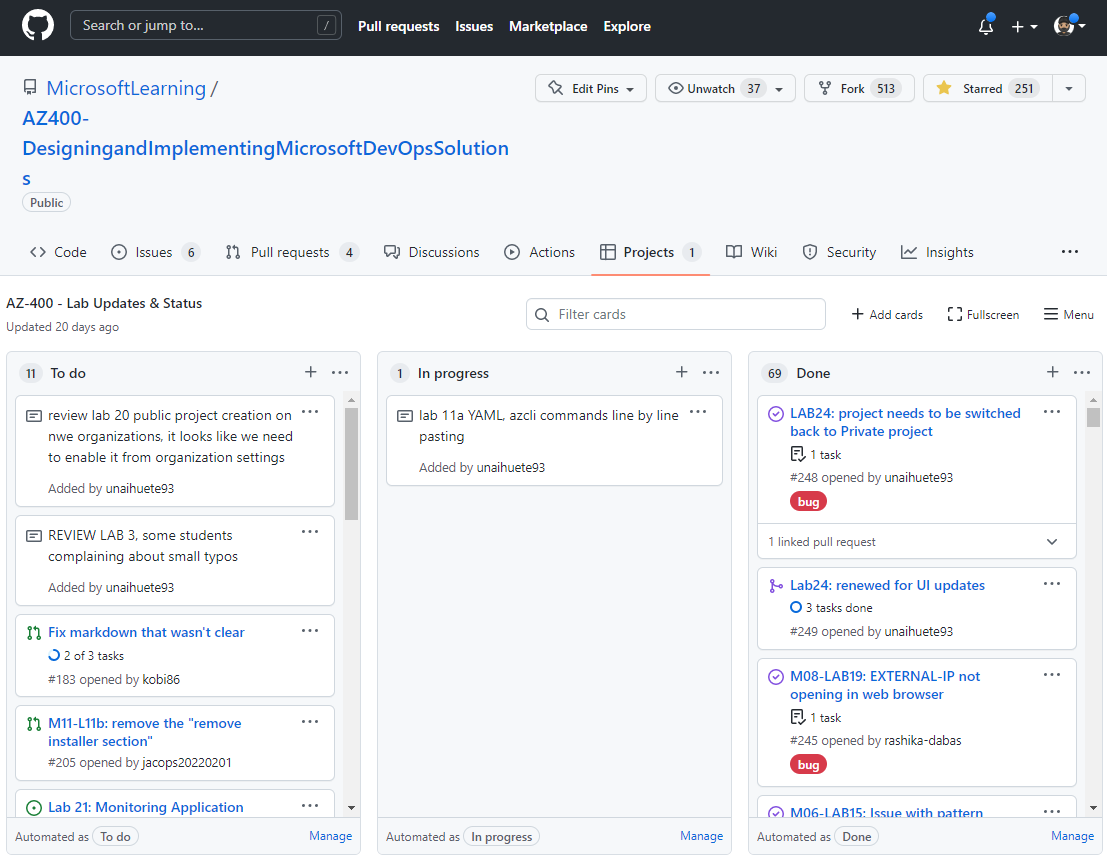
Completed100 XP

* 3 minutes

**Project Boards**

During the application or project lifecycle, it's crucial to plan and prioritize work. With Project boards, you can control specific feature work, roadmaps, release plans, etc.

Project boards are made up of issues, pull requests, and notes categorized as cards you can drag and drop into your chosen columns. The cards contain relevant metadata for issues and pull requests, like labels, assignees, the status, and who opened it.



There are different types of project boards:

* **User-owned project boards:** Can contain issues and pull requests from any personal repository.
* **Organization-wide project boards:** Can contain issues and pull requests from any repository that belongs to an organization.
* **Repository project boards:** Are scoped to issues and pull requests within a single repository.

To create a project board for your organization, you must be an organization member.

It's possible to use templates to set up a new project board that will include columns and cards with tips. The templates can be automated and already configured.

Expand table

| **Templates** | **Description** |
| --- | --- |
| Basic kanban | Track your tasks with: To do, In progress, and Done columns. |
| Automated kanban | Cards automatically move between: To do, In progress, and Done columns. |
| Automated kanban with review | Cards automatically move between: To do, In progress, and Done columns, with extra triggers for pull request review status. |
| Bug triage | Triage and prioritize bugs with: To do, High priority, Low priority, and Closed columns. |

For more information about Project boards, see:

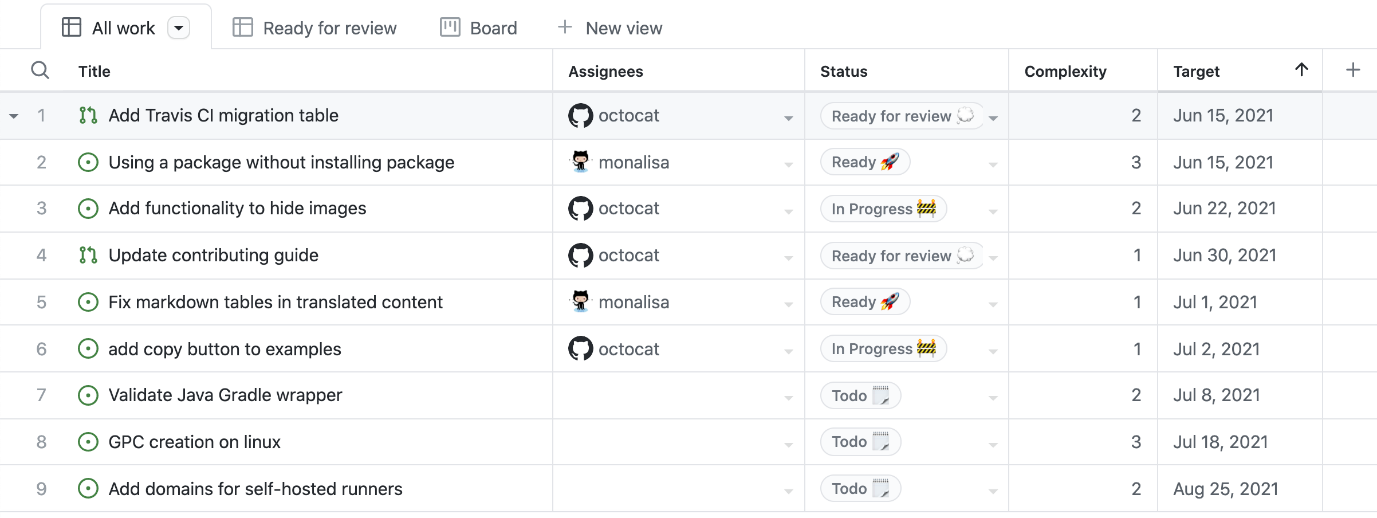
* [Creating a project board](https://docs.github.com/articles/creating-a-project-board).
* [Editing a project board](https://docs.github.com/articles/editing-a-project-board).
* [Copying a project board](https://docs.github.com/issues/planning-and-tracking-with-projects/creating-projects/copying-an-existing-project).
* [Adding issues and pull requests to a project board](https://docs.github.com/articles/adding-issues-and-pull-requests-to-a-project-board).
* [Project board permissions for an organization](https://docs.github.com/articles/project-board-permissions-for-an-organization).

**Projects**

Projects are a new, customizable and flexible tool version of projects for planning and tracking work on GitHub.

A project is a customizable spreadsheet in which you can configure the layout by filtering, sorting, grouping your issues and PRs, and adding custom fields to track metadata.

You can use different views such as Board or spreadsheet/table.



If you change your pull request or issue, your project reflects that change.

You can use custom fields in your tasks. For example:

* A date field to track target ship dates.
* A number field to track the complexity of a task.
* A single select field to track whether a task is Low, Medium, or High priority.
* A text field to add a quick note.
* An iteration field to plan work week-by-week, including support for breaks.

For more information about Projects, see:

* [Creating a project](https://docs.github.com/issues/trying-out-the-new-projects-experience/creating-a-project).
* [Managing iterations in projects](https://docs.github.com/issues/trying-out-the-new-projects-experience/managing-iterations).
* [Customizing your project views](https://docs.github.com/issues/trying-out-the-new-projects-experience/customizing-your-project-views).
* [Automating projects](https://docs.github.com/issues/trying-out-the-new-projects-experience/automating-projects).

**Next unit: Introduction to Azure Boards**

**Introduction to Azure Boards**

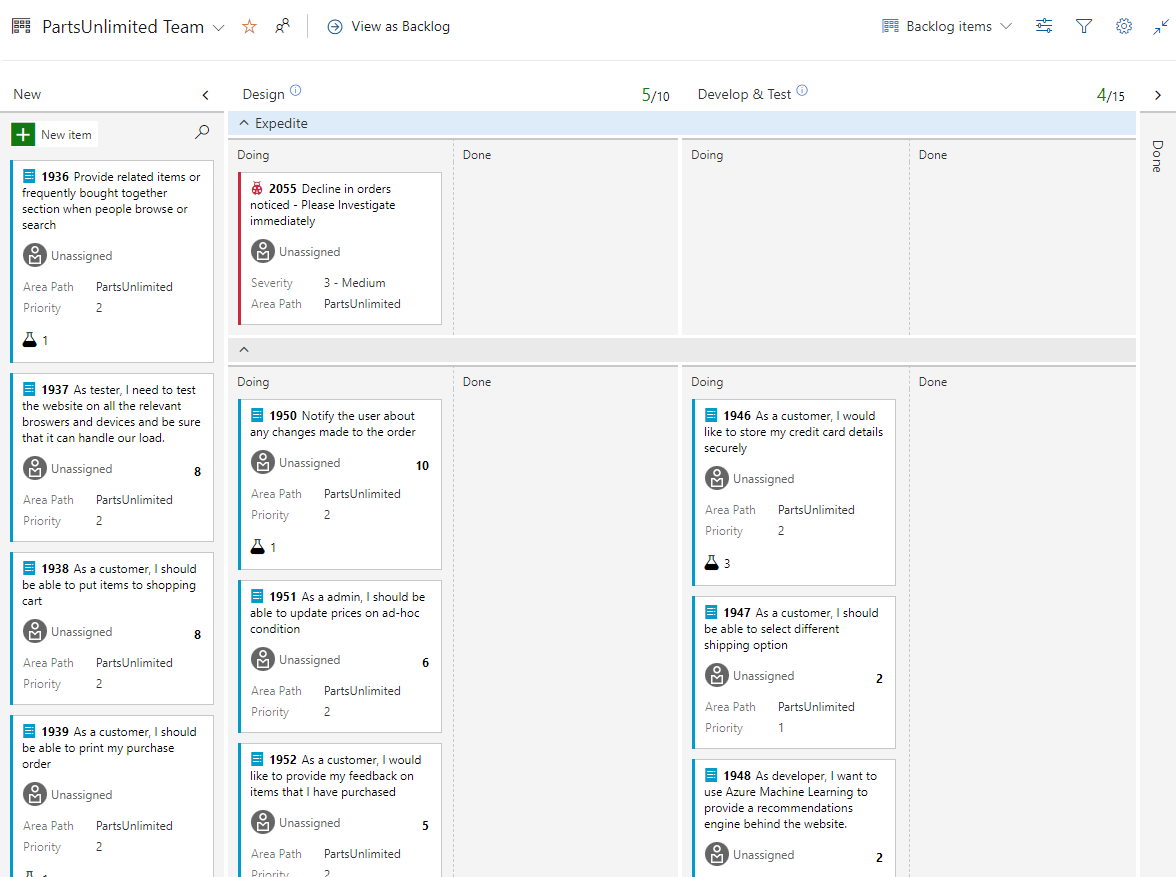
Completed100 XP

* 3 minutes

Azure Boards is a customizable tool to manage software projects supporting Agile, Scrum, and Kanban processes by default. Track work, issues, and code defects associated with your project. Also, you can create your custom process templates and use them to create a better and more customized experience for your company.

You have multiple features and configurations to support your teams, such as calendar views, configurable dashboards, and integrated reporting.

The Kanban board is one of several tools that allows you to add, update, and filter user stories, bugs, features, and epics.



You can track your work using the default work item types such as user stories, bugs, features, and epics. It's possible to customize these types or create your own. Each work item provides a standard set of system fields and controls, including Discussion for adding and tracking comments, History, Links, and Attachments.

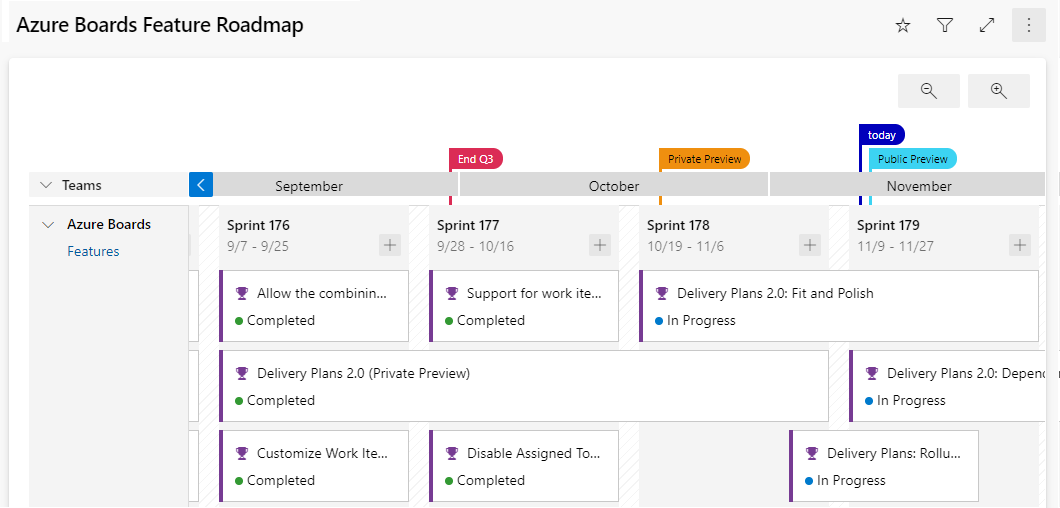
If you need to create reports or a list of work with specific filters, you can use the queries hub to generate custom lists of work items.

Queries support the following tasks:

* Find groups of work items with something in common.
* Triage work to assign to a team member or sprint and set priorities.
* Perform bulk updates.
* View dependencies or relationships between work items.
* Create status and trend charts that you can optionally add to dashboards.

**Delivery plans**

It's possible to create another view with deliverables and track dependencies across several teams in a calendar view using Delivery Plans.



Delivery plans are fully interactive, supporting the following tasks:

* View up to 15 team backlogs, including a mix of backlogs and teams from different projects.
* View custom portfolio backlogs and epics.
* View work that spans several iterations.
* Add backlog items from a plan.
* View rollup progress of features, epics, and other portfolio items.
* View dependencies that exist between work items.

For more information about Azure Boards, see:

* [Azure Boards documentation | Microsoft Learn](https://learn.microsoft.com/en-us/azure/devops/boards).
* [Reasons to start using Azure Boards | Microsoft Learn](https://learn.microsoft.com/en-us/azure/devops/boards/get-started/why-use-azure-boards).
* [GitHub and Azure Boards](https://learn.microsoft.com/en-us/azure/devops/boards/github).

**Next unit: Configure projects and teams in Azure DevOps**

**Configure projects and teams in Azure DevOps**

Completed100 XP

* 3 minutes

In today's highly dynamic software development landscape, efficient project organization and team collaboration are essential components of DevOps strategy. Projects and teams in Azure DevOps form the foundation for efficient communication, clear accountability, and streamlined workflow management. They also contribute towards enhanced agility, accelerated delivery, and proper alignment with business objectives.

**Planning**

To build the optimal approach to implementing projects and teams, you should first analyze the organizational structure, its operational model, and the roles that contribute to the development lifecycle. This analysis should consider such factors as departmental divisions, current business initiatives, and any existing projects that might introduce potential cross-project dependencies.

Next, for each new project, determine its scope, objectives, and stakeholders. Based on the organizational model and project requirements, identify the optimal way to structure the project's teams. Decide whether teams will be cross-functional or organized based on such criteria as features and components. Identify the roles and responsibilities within the project teams. Common roles include developers, testers, product owners, Scrum masters, and project managers. Clearly define the expectations and responsibilities associated with each role. Ensure sufficient techniques and methods for collaboration and communication within and between teams.

Establish governance policies and guidelines for project management, including version control policies, branching strategies, code review processes, and verification of compliance requirements. Define the agile processes and methodologies that will be used for project management. Keep in mind that you can customize the Agile tools in Azure DevOps to support your decision.

To streamline the process of creating new projects, consider defining templates or blueprints that define project structure, processes, and artifacts. Such templates might include predefined settings of source code repositories, work item tracking, pipelines, and test plans.

**Implementation**

Once you complete all the preparations, proceed with implementing the planned project. At the very outset, you need to specify the project's visibility (public or private), decide between the Git and Team Foundation Version Control as the version control mechanism, and choose the work item process. That choice includes the following options:

* **Agile:** Focuses on iterative development and delivering customer value through user stories, backlogs, sprints, and visual boards for tracking progress.
* **Basic:** Offers a simple and flexible approach with generic work items and basic backlog and board functionalities, most suitable for small teams or projects.
* **CMMI (Capability Maturity Model Integration):** Provides formalized processes and standards for managing work items, requirements, and quality assurance, which is most suitable for organizations seeking process improvement and compliance.
* **Scrum:** Employs an iterative, self-organizing framework with product and sprint backlogs, daily stand-ups, and artifacts for managing work and delivering incremental changes.
* **Customized Scrum:** Facilitates tailoring of the Scrum process to fit specific needs through flexibility in work item types, fields, and workflow states.

With the project in place, you can proceed to building its team structure. By default, every project includes one team which name is derived from the project name, so, depending on the outcome of your planning, you might need to modify it. You might also want to create additional teams. Each team must have at least one administrator. Each team administrator can add team members.

When creating a team, it is also possible to automatically generate an area path with the matching name. Area paths help organize work items within a project, facilitating better visibility, reporting, and access control. Associating individual teams with a dedicated area path helps clearly delineate between their areas of responsibility.

Once the teams are created, you can configure for each of them a wide range of additional settings, including customized permissions. By default, team members are added to the Contributors group on the project level. At this point, you can also start managing project resources such as repositories, boards, pipelines, and test plans and assign them to specific teams as required.

In the spirit of continuous improvement, you should regularly review and refine the project and team configuration based on feedback from teams' and project's members. This will ensure that your organization can efficiently manage the process of developing software products regardless of their complexity.

**Next unit: Link GitHub to Azure Boards**

**Link GitHub to Azure Boards**

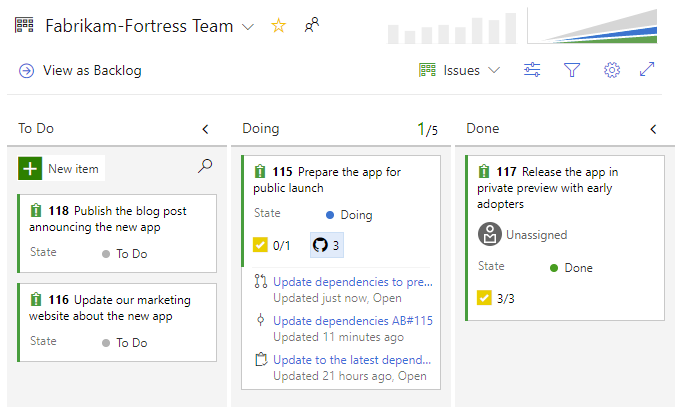
Completed100 XP

* 2 minutes

**Use GitHub, track work in Azure Boards**

Use Azure Boards to plan and track your work and GitHub as source control for software development.

Connect Azure Boards with GitHub repositories, enabling linking GitHub commits, pull requests, and issues to work items in Boards.

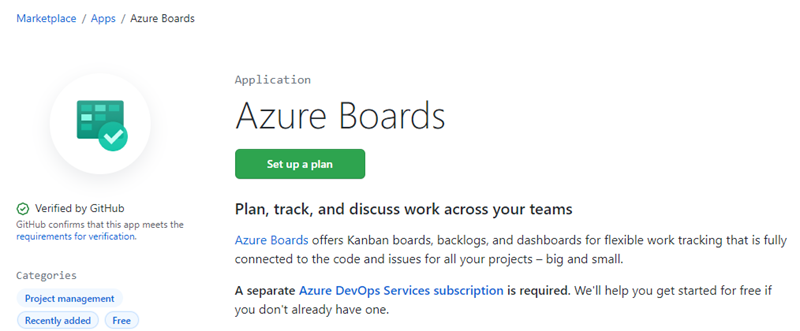


**Azure Boards App**

The integration is created using the Azure Boards App, acting as a bridge between Azure Boards and GitHub.

To install the app, you must be an administrator or owner of the GitHub repository or the GitHub organization.

The app is installed from the GitHub Marketplace. [Azure Boards App](https://github.com/marketplace/azure-boards)



**Authenticating to GitHub**

Azure Boards can connect to GitHub. For GitHub in the cloud, when adding a GitHub connection, the authentication options are:

* Username/Password
* Personal Access Token (PAT)

For a walkthrough on making the connection, see: [Connect Azure Boards to GitHub](https://learn.microsoft.com/en-us/azure/devops/boards/github/connect-to-github).

You can configure other Azure Boards/Azure DevOps Projects, GitHub.com repositories, or change the current configuration from the Azure Boards app page.

Once you've integrated Azure Boards with GitHub using the Azure Boards app, you can add or remove repositories from the web portal for Azure Boards.

**Supported integration scenarios**

Azure Boards-GitHub integration supports the following connections:

* From GitHub:
  + Support integration for all repositories for a GitHub account or organization or select repositories.
  + Add or remove GitHub repositories participating in the integration and configure the project they connect to.
  + Suspend Azure Boards-GitHub integration or uninstall the app.
* From Azure Boards:
  + Connect one or more GitHub repositories to an Azure Boards project.
  + Add or remove GitHub repositories from a GitHub connection within an Azure Boards project.
  + Completely remove a GitHub connection for a project.
  + Allow a GitHub repository to connect to one or more Azure Boards projects within the same Azure DevOps organization or collection.

Azure Boards-GitHub integration supports the following operational tasks:

* Create links between work items and GitHub commits, pull requests, and issues based on GitHub mentions.
* Support state transition of work items to a Done or Completed state when using GitHub mention by using fix, fixes, or fixed.
* Support full traceability by posting a discussion comment to GitHub when linking from a work item to a GitHub commit, pull request, or issue.
* Show linked to GitHub code artifacts within the work item Development section.
* Show linked to GitHub artifacts as annotations on Kanban board cards.
* Support status badges of Kanban board columns added to GitHub repositories.

The following tasks aren't supported at this time:

* Query for work items with links to GitHub artifacts. However, you can query for work items with an External Link Count greater than 0.

**Note**

Reference: [**Azure Boards-GitHub integration**](https://learn.microsoft.com/en-us/azure/devops/boards/github).

For more information, see:

* [Change GitHub repository access, or suspend or uninstall the integration](https://learn.microsoft.com/en-us/azure/devops/boards/github/change-azure-boards-app-github-repository-access).
* [Add or remove GitHub repositories](https://learn.microsoft.com/en-us/azure/devops/boards/github/add-remove-repositories).
* [Link GitHub commits, pull requests, and issues to work items for details on linking to work items](https://learn.microsoft.com/en-us/azure/devops/boards/github/link-to-from-github).

**Next unit: Configure GitHub Projects**

**Link GitHub to Azure Boards**

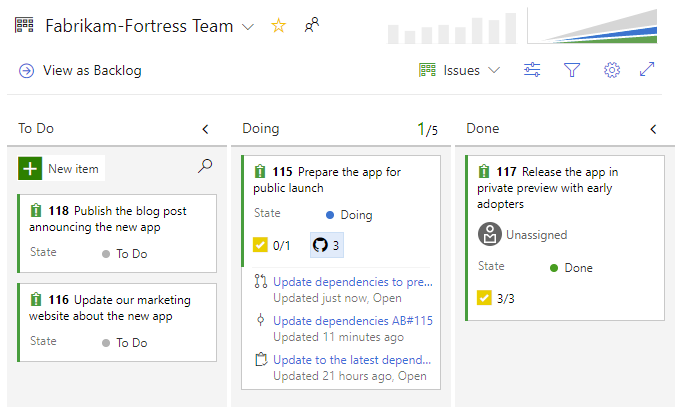
Completed100 XP

* 2 minutes

**Use GitHub, track work in Azure Boards**

Use Azure Boards to plan and track your work and GitHub as source control for software development.

Connect Azure Boards with GitHub repositories, enabling linking GitHub commits, pull requests, and issues to work items in Boards.

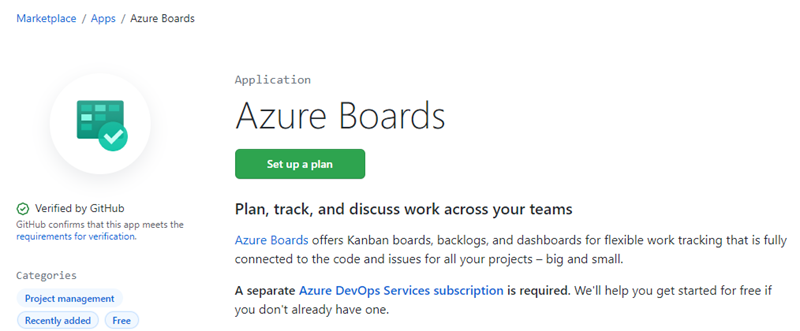


**Azure Boards App**

The integration is created using the Azure Boards App, acting as a bridge between Azure Boards and GitHub.

To install the app, you must be an administrator or owner of the GitHub repository or the GitHub organization.

The app is installed from the GitHub Marketplace. [Azure Boards App](https://github.com/marketplace/azure-boards)



**Authenticating to GitHub**

Azure Boards can connect to GitHub. For GitHub in the cloud, when adding a GitHub connection, the authentication options are:

* Username/Password
* Personal Access Token (PAT)

For a walkthrough on making the connection, see: [Connect Azure Boards to GitHub](https://learn.microsoft.com/en-us/azure/devops/boards/github/connect-to-github).

You can configure other Azure Boards/Azure DevOps Projects, GitHub.com repositories, or change the current configuration from the Azure Boards app page.

Once you've integrated Azure Boards with GitHub using the Azure Boards app, you can add or remove repositories from the web portal for Azure Boards.

**Supported integration scenarios**

Azure Boards-GitHub integration supports the following connections:

* From GitHub:
  + Support integration for all repositories for a GitHub account or organization or select repositories.
  + Add or remove GitHub repositories participating in the integration and configure the project they connect to.
  + Suspend Azure Boards-GitHub integration or uninstall the app.
* From Azure Boards:
  + Connect one or more GitHub repositories to an Azure Boards project.
  + Add or remove GitHub repositories from a GitHub connection within an Azure Boards project.
  + Completely remove a GitHub connection for a project.
  + Allow a GitHub repository to connect to one or more Azure Boards projects within the same Azure DevOps organization or collection.

Azure Boards-GitHub integration supports the following operational tasks:

* Create links between work items and GitHub commits, pull requests, and issues based on GitHub mentions.
* Support state transition of work items to a Done or Completed state when using GitHub mention by using fix, fixes, or fixed.
* Support full traceability by posting a discussion comment to GitHub when linking from a work item to a GitHub commit, pull request, or issue.
* Show linked to GitHub code artifacts within the work item Development section.
* Show linked to GitHub artifacts as annotations on Kanban board cards.
* Support status badges of Kanban board columns added to GitHub repositories.

The following tasks aren't supported at this time:

* Query for work items with links to GitHub artifacts. However, you can query for work items with an External Link Count greater than 0.

**Note**

Reference: [**Azure Boards-GitHub integration**](https://learn.microsoft.com/en-us/azure/devops/boards/github).

For more information, see:

* [Change GitHub repository access, or suspend or uninstall the integration](https://learn.microsoft.com/en-us/azure/devops/boards/github/change-azure-boards-app-github-repository-access).
* [Add or remove GitHub repositories](https://learn.microsoft.com/en-us/azure/devops/boards/github/add-remove-repositories).
* [Link GitHub commits, pull requests, and issues to work items for details on linking to work items](https://learn.microsoft.com/en-us/azure/devops/boards/github/link-to-from-github).

**Next unit: Configure GitHub Projects**

**Manage work with GitHub Project boards**

Completed100 XP

* 2 minutes

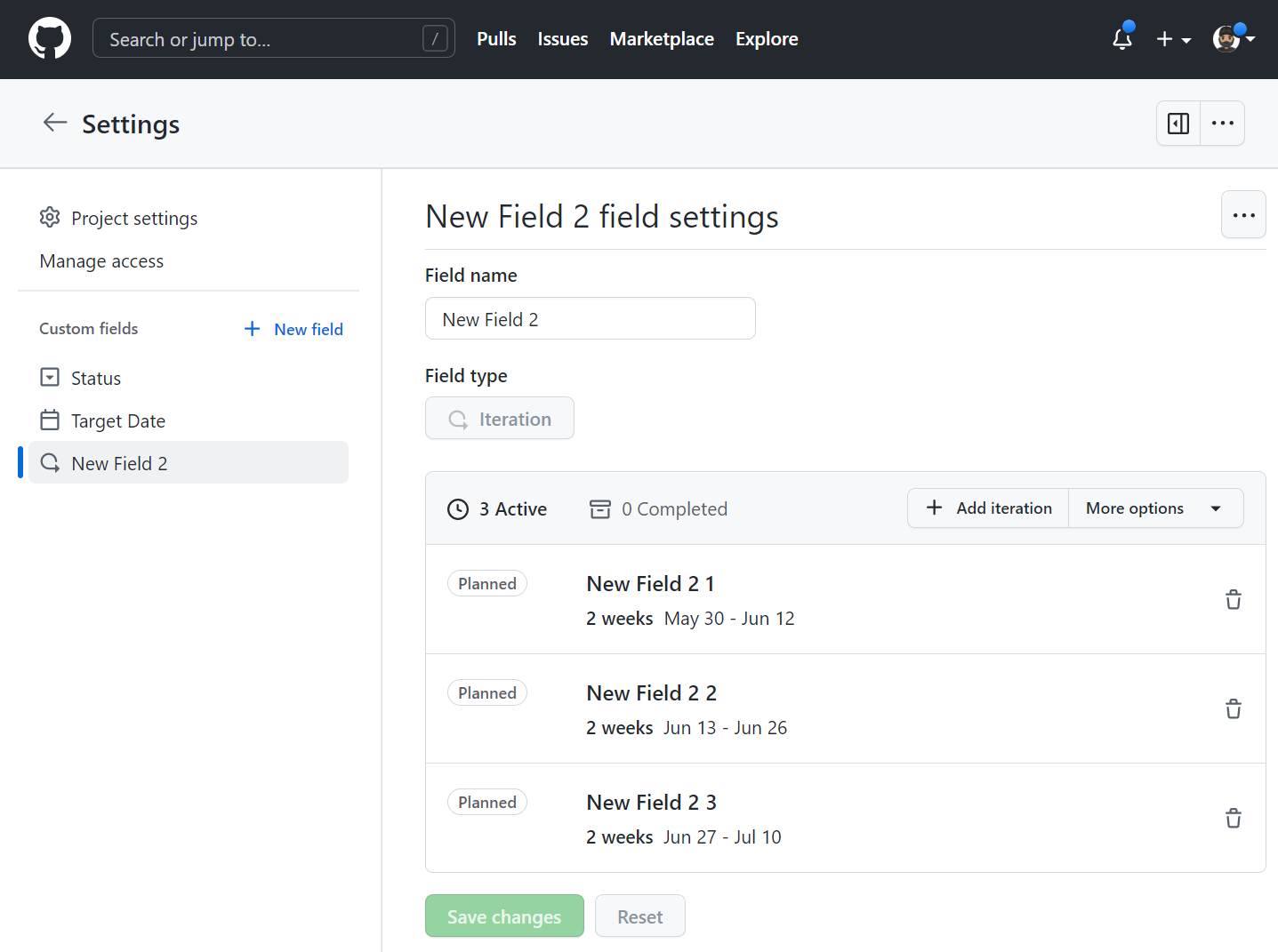
GitHub Projects allow you to control project deliverables, release dates, and iterations to plan upcoming work.

You can create an iteration to:

* Associate items with specific repeating blocks of time.
* Set to any length of time.
* Include breaks.

It's possible to configure your project to group by iteration to visualize the balance of upcoming work.

When you first create an iteration field, three iterations are automatically created. You can add other iterations if needed.



**Iteration field**

You can use the command palette or the project's interface to create an iteration field.

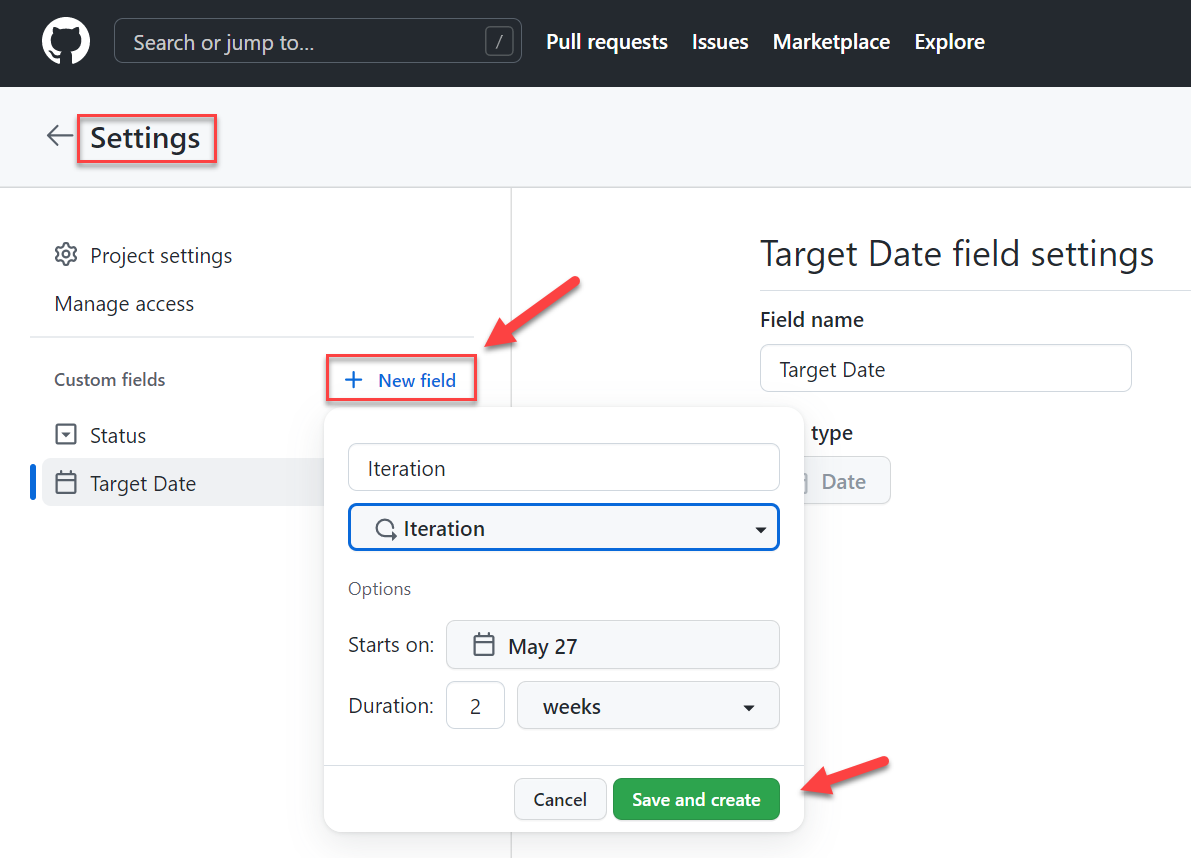
**Tip**

To open the project command palette, press Ctrl+K (Windows/Linux) or Command+K (Mac).

Start typing any part of "Create new field". When "Create new field" displays in the command palette, select it.

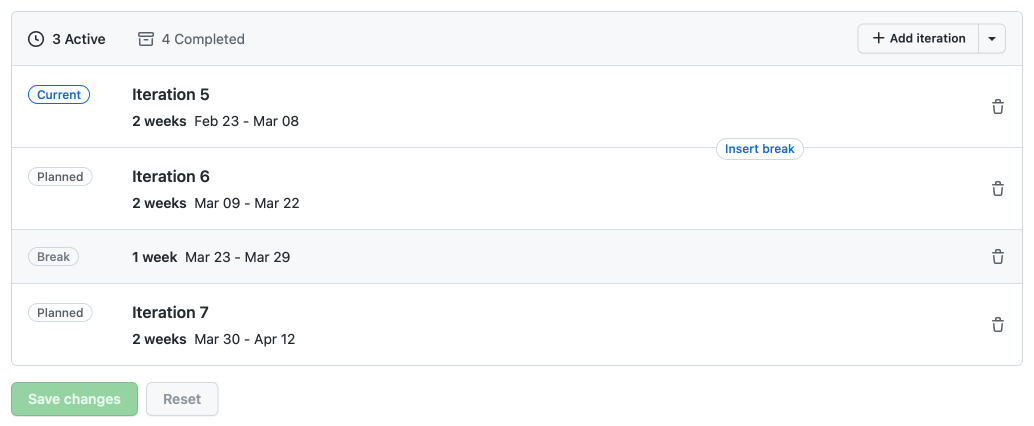
Or follow the steps using the interface:

1. Navigate to your project.
2. Click in the plus (**+**) sign in the rightmost field header. A drop-down menu with the project fields will appear.
3. Click in the **New field**.
4. Enter a name for the new iteration field.
5. Select the dropdown menu below and click Iteration.
6. (Optional) Change the starting date from the current day, select the calendar dropdown next to Starts on, and click on a new starting date.
7. To change the duration of each iteration, type a new number, then select the dropdown and click either days or weeks.
8. Click Save and create.



**Adding new iterations**

1. Navigate to your project.
2. In the top-right, click to open the menu.
3. In the menu, click **Settings** to access the project settings.
4. Click the name of the iteration field you want to adjust.
5. To add a new iteration of the same duration, click **Add iteration**.
6. (Optional) Customize the duration of the new iteration and when it starts.
   1. Click next to Add iteration.
   2. Select a starting date and duration.
   3. Click **Add**.
7. Click Save changes.



Also, you can insert breaks into your iterations to communicate when you're taking time away from scheduled work.

For more information about iterations, see:

* [Managing iterations in projects (beta) - GitHub Docs](https://docs.github.com/issues/trying-out-the-new-projects-experience/managing-iterations).
* [Best practices for managing projects (beta) - GitHub Docs](https://docs.github.com/issues/trying-out-the-new-projects-experience/best-practices-for-managing-projects).

**Next unit: Customize Project views**

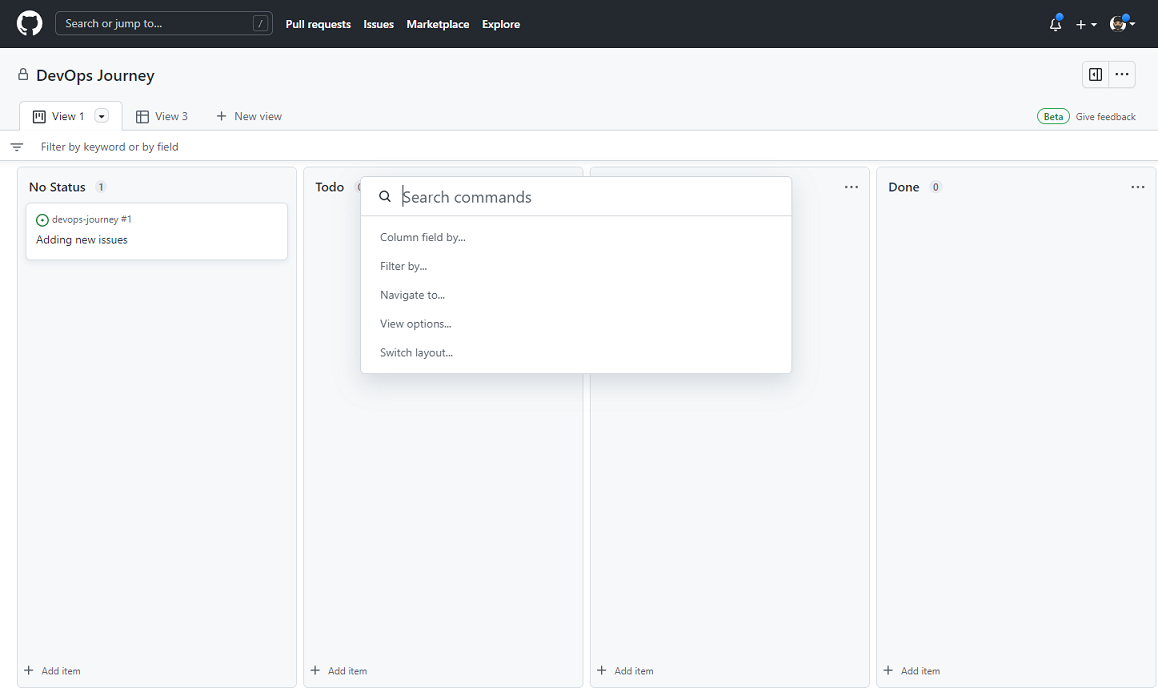
**Customize Project views**

Completed100 XP

* 3 minutes

Using Projects views, you can organize information by changing the layout, grouping, sorting, and filtering your work.

You can create and use different visualizations, for example, Board view:



**Project command palette**

Use the project command palette to change settings and run commands in your project.

1. Open the project command palette by pressing **Command + K (Mac)** or **Ctrl + K (Windows/Linux)**.
2. Type any command part or navigate through the command palette window to find a command.

You have multiple commands to apply, such as:

* Switch layout: Table.
* Show: Milestone.
* Sort by: Assignees, asc.
* Remove sort-by.
* Group by: Status.
* Remove group-by.
* Column field by: Status.
* Filter by Status.
* Delete view.

**Note**

For more information about GitHub Command Palette, see [**GitHub Command Palette - GitHub Docs**](https://docs.github.com/get-started/using-github/github-command-palette).

Also, you can perform changes using the interface.

**Creating a project view**

Project views allow you to view specific aspects of your project. Each view is displayed on a separate tab in your project.

For example, you can have:

* A view that shows all items not yet started (filter on Status).
* A view that shows the workload for each team (group by a custom Team field).
* A view that shows the items with the earliest target ship date (sort by a date field).

To add a new view:

1. To open the project command palette, press **Command + K (Mac)** or **Ctrl + K (Windows/Linux)**.
2. Start typing **New view** (to create a new view) or **Duplicate view** (to duplicate the current view).
3. Choose the required command.
4. The new view is automatically saved.

For more information about projects (beta), see:

* [About projects (beta)](https://docs.github.com/issues/trying-out-the-new-projects-experience/about-projects).
* [Creating a project (beta)](https://docs.github.com/issues/trying-out-the-new-projects-experience/creating-a-project).

**Next unit: Collaborate using team discussions**

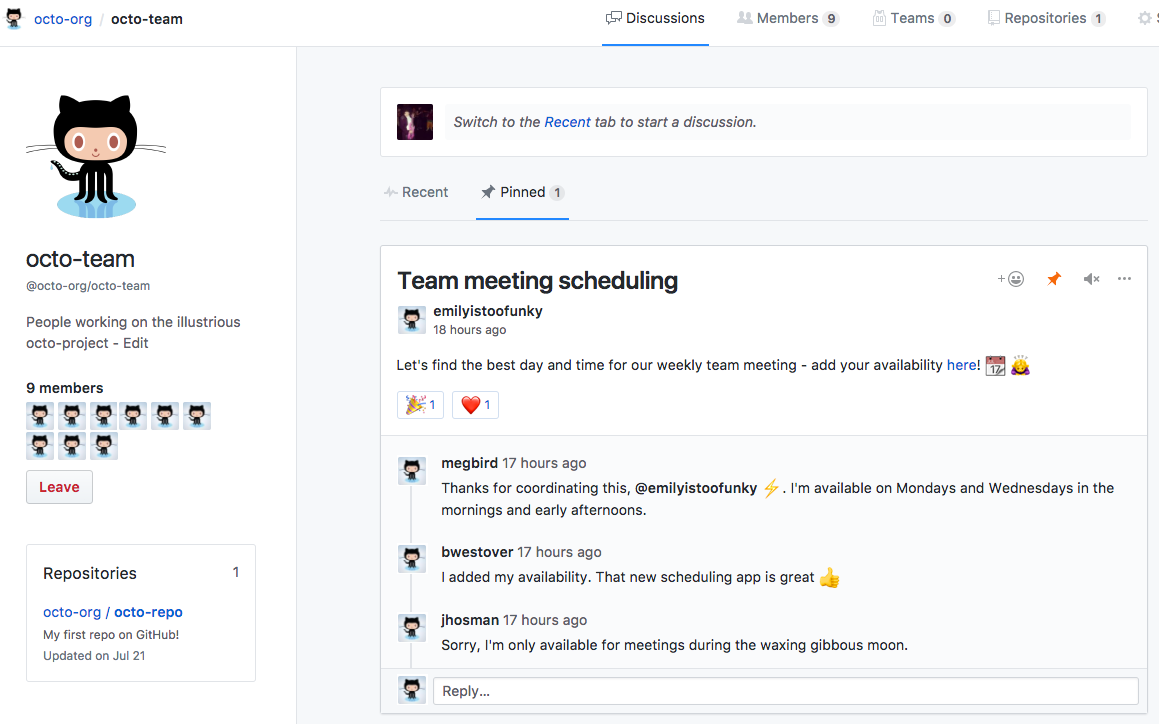
**Collaborate using team discussions**

Completed100 XP

* 3 minutes

GitHub discussions can help make your team plan together, update one another, or talk about any topic you'd like in discussion posts on your team's page in an organization.

You can use team discussions for conversations that span across projects or repositories (issues, pull requests, etc.). Instead of opening an issue in a repository to discuss an idea, you can include the entire team by having a conversation in a team discussion.



With team discussions, you can:

* Post on your team's page or participate in a public discussion.
* Link to any team discussion to reference it elsewhere.
* Pin important posts to your team's page.
* Receive email or web notifications.

Team discussions are available in organizations by default.

You can also use organization discussions to facilitate conversations across your organization.

For more information about team discussion, see:

* [Enabling or disabling GitHub Discussions for an organization](https://docs.github.com/organizations/managing-organization-settings/enabling-or-disabling-github-discussions-for-an-organization).
* [Quickstart for communicating on GitHub](https://docs.github.com/github/collaborating-with-issues-and-pull-requests/quickstart-for-communicating-on-github).
* [About teams](https://docs.github.com/articles/about-teams).
* [Creating a team discussion](https://docs.github.com/organizations/collaborating-with-your-team/creating-a-team-discussion).
* [Editing or deleting a team discussion](https://docs.github.com/organizations/collaborating-with-your-team/editing-or-deleting-a-team-discussion).

**Next unit: Design and implement a strategy for feedback cycles**

**Design and implement a strategy for feedback cycles**

Completed100 XP

* 3 minutes

The continuing success of practically every software product is contingent on the ability to assess its state from any stage of its lifecycle. Feedback cycles deliver the data required for this assessment, promoting continuous improvement. Robust notification systems ensure that teams developing and supporting the product have up-to-date information to act on and can respond in a timely manner. Effective issue management facilitates swift resolution through effective tracking and triage mechanisms.

**Design**

Designing a strategy for feedback cycles, notifications, and issue management in Azure DevOps starts with defining channels for receiving feedback. This involves identifying sources such as user feedback forms, customer support tickets, bug reports, and automated testing results, and establishing processes to collect, organize, and prioritize this feedback.

Notification rules play a crucial role in keeping team members informed about relevant software lifecycle events. The rules generate alerts for new work items, build failures, code reviews, and pull request approvals. They are customizable and can be tailored based on project requirements and individual preferences.

Additionally, Azure DevOps supports integration of its internal feedback mechanisms with several external feedback delivery tools, which streamlines the process of aggregating relevant information from multiple sources. Customizable review and triage processes further enhance the efficiency of feedback management by supporting a range of prioritization criteria, including severity, impact, and customer value.

**Implementation**

The implementation phase involves setting up notification rules within Azure DevOps to ensure that relevant stakeholders are promptly informed about key events and updates. This includes configuring notification settings based on team roles and project requirements.

Issue tracking and management systems can be implemented by using Azure Boards. Work item types are created to categorize different types of feedback, such as bugs, tasks, and user stories. Workflows provide guidance with issue resolution.

Integration with external feedback tools and systems is typically facilitated through Azure DevOps extensions, allowing for seamless data synchronization and centralized feedback management. This could include linking to customer feedback portals, helpdesk systems, and third-party testing tools.

Another important aspect of implementation is tracking of key performance indicators related to feedback cycles. Most followed indicators include response time, resolution time, customer satisfaction scores, and defect density. Regular reviews of feedback metrics assist with establishing baselines, identifying trends, and detecting anomalies indicative of emerging issues. In addition, such reviews help identify opportunities for process optimization, product enhancements, and quality improvements. Azure DevOps Analytics and reporting capabilities considerably simplify reaching these objectives. Similarly, retrospectives and post-mortems foster reviews of feedback cycles and lessons learned, leading to corrective actions that minimize the possibility of repeating past mistakes.

**Next unit: Design and implement source, bug, and quality traceability**

**Design and implement source, bug, and quality traceability**

Completed100 XP

* 3 minutes

In the context of DevOps, traceability refers to the ability to track changes and actions throughout the software development lifecycle. This concept applies to different aspects of that lifecycle, including changes to source code, bug resolution, and maintaining quality control. Its implementation is essential to ensure product reliability, maintainability, and customer satisfaction.

Source traceability enables developers to track code changes in collaboration scenarios. Bug traceability facilitates identifying, prioritizing, and resolving issues with the source code. Quality traceability ensures that software meets quality standards and user expectations by linking testing activities, metrics, and feedback to development efforts.

**Design**

At a high level, traceability is tool-agnostic but the way to approach it depends on the aspect of software development lifecycle it is meant to target. Similarly, objectives and design considerations differ between source code, bug, and quality traceability.

In particular, source traceability involves tracking the history of code changes, including who made the changes, when they were made, and the purpose of the changes. It facilitates code reviews, debugging, and understanding the evolution of the codebase over time. From the design standpoint, this functionality ties closely to Git branching and merging strategies that organize development work. Developers create feature branches for new work, commit changes to their branches, and submit pull requests for review. At that point, their peers conduct code reviews and, once successfully completed, approve the changes to be merged into the main branch.

Bug traceability involves tracing bugs or defects reported during testing or production back to their root cause in the codebase. It also commonly relies on capturing information such as the bug report details, steps to reproduce, affected components, and related code changes. Its objectives include prioritizing and resolving bugs efficiently in order to address software deficiencies.

Quality traceability encompasses tracing quality-related activities and artifacts throughout the software development process. This involves linking quality metrics, test cases, test results, and other quality assurance activities to requirements, user stories, and code changes. Quality traceability helps assess the impact of software changes on its quality and identify areas for improvement.

**Implementing traceability**

Traceability implementation details of differ to some extent depending on the DevOps platform.

**Source traceability**

Since both GitHub and Azure DevOps support Git as their source control mechanism, a number of source traceability techniques apply to both of them. As the result, implementing source code traceability in both cases involves adopting best practices such as writing descriptive commit messages, using a well-defined branching strategy, and requiring pull requests for code reviews.

However, there are also some differences between them. Implementing source traceability in GitHub repositories typically involves utilizing features such as branch protection rules to enforce code review processes and ensure changes are reviewed before merging. GitHub's integration with Issues allows for linking code changes to corresponding issues, providing traceability between code modifications and project requirements. Azure DevOps offers branch policies and pull request policies for enforcing code quality checks and linking changes to work items, enabling traceability between code changes and user stories or tasks. Additionally, Azure DevOps provides more extensive integration with its work item tracking system, allowing for deeper traceability and reporting capabilities compared to GitHub's issue tracking.

**Bug traceability**

In Azure DevOps, bug traceability is facilitated through Azure Boards, where bugs are tracked as work items and can be linked to code changes, commits, and pull requests. Azure Boards allows for creating custom workflows for bug management, defining states such as New, Active, Resolved, and Closed, providing visibility into the bug's lifecycle. Additionally, Azure DevOps offers rich integration between bugs and other work items, enabling traceability between bugs and user stories, tasks, and epics.

In GitHub, bug traceability relies on the integration between issues and code changes, where bugs reported as issues can be linked to commits and pull requests. GitHub Actions offer the ability to implement customizable workflows, including those related to bug traceability. With GitHub Actions, you can define workflows that automate processes triggered by events in your GitHub repository, such as the creation or modification of issues. This allows you to create custom workflows for managing bugs, including defining states, assigning tasks, and automating actions based on specific conditions. Effectively, while GitHub Actions provide flexibility in workflow automation, they typically require more effort and customization compared to the built-in features of Azure Boards in Azure DevOps.

**Quality traceability**

In Azure DevOps, quality traceability can be managed using Test Plans, which allow teams to organize, execute, and track test cases. Test Plans provide comprehensive quality metrics, including test case pass rates, test run outcomes, and test coverage reports. Additionally, Azure DevOps offers integration with code coverage tools to measure test coverage and identify areas of the codebase that require additional testing.

GitHub offers similar functionality through GitHub Actions, enabling teams to automate various types of tests, such as unit tests, integration tests, and end-to-end tests. Here as well, GitHub Actions provides flexibility in setting up test workflows and integrating with third-party testing tools, but they tend to require additional configuration to achieve the same level of comprehensive quality metrics and test coverage reporting as Azure DevOps Test Plans.

**Next unit: Agile Plan and Portfolio Management with Azure Boards**

**Agile Plan and Portfolio Management with Azure Boards**

Completed100 XP

* 57 minutes

**Estimated time:** 60 minutes.

**Scenario**

In this lab, you'll learn about the agile planning and portfolio management tools and processes provided by Azure Boards and how they can help you quickly plan, manage, and track work across your entire team. You'll explore the product backlog, sprint backlog, and task boards that can track the flow of work during an iteration. We'll also look at the enhanced tools in this release to scale for larger teams and organizations.

**Objectives**

After completing this lab, you'll be able to:

* Manage teams, areas, and iterations.
* Manage work items.
* Manage sprints and capacity.
* Customize Kanban boards.
* Define dashboards.
* Customize team process.

**Requirements**

* This lab requires **Microsoft Edge** or an [Azure DevOps-supported browser](https://learn.microsoft.com/en-us/azure/devops/server/compatibility).
* **Set up an Azure DevOps organization:** If you don't already have an Azure DevOps organization that you can use for this lab, create one by following the instructions available at [Create an organization or project collection](https://learn.microsoft.com/en-us/azure/devops/organizations/accounts/create-organization).

**Exercises**

During this lab, you'll complete the following exercises:

* Exercise 0: Configure the lab prerequisites.
* Exercise 1: Manage an Agile project.
* Exercise 2 (optional): Define dashboards.

[Screenshot of a launch button which will take you to the lab.](https://go.microsoft.com/fwlink/?linkid=2269557)

**Next unit: Knowledge check**

**Knowledge check**

Completed200 XP

* **Module assessment**
* 3 minutes

 Answer 100% of questions correctly in order to pass. [**Retake**](https://learn.microsoft.com/en-us/training/modules/plan-agile-github-projects-azure-boards/13-knowledge-check)

Dismiss alert

Choose the best response for each question.

**Check your knowledge**

Top of Form

**1.**

**Which of the following choices best describes the Azure DevOps and GitHub integration?**

Azure Boards have direct integration with Azure Repos, but it can also be integrated with GitHub to plan and track work linking commits, PRs, and issues.

**Correct. Integrating GitHub with Azure Boards lets you plan and track your work by linking GitHub commits, pull requests, and issues, directly to work items in Boards.**

Azure Boards has direct integration with Azure Repos. Azure Repos only integrates with GitHub using extensions from Marketplace for the read-only track.

Azure Boards has direct integration with GitHub for tracking activities and moving tasks on both sides Azure Boards to GitHub and GitHub to Azure Repos.

**Incorrect. Integrating GitHub with Azure Boards lets you plan and track your work by linking GitHub commits, pull requests, and issues, directly to work items in Boards.**

**2.**

**Which of the following Project boards types can contain issues and pull requests from any personal repository?**

User-owned project boards.

**Correct. User-owned project boards can contain issues and pull requests from any personal repository.**

Organization-wide project boards.

**Incorrect. Organization-wide project boards can contain issues and pull requests from any repository that belongs to an organization.**

Repository project boards

**3.**

**Which of the following choices isn't a Project Boards supported template by default?**

Basic kanban.

Automated kanban with review.

**Incorrect. Automated kanban with review is a supported template that cards automatically move between: To do, In progress, and Done columns, with extra triggers for pull request review status.**

Automated CMMI.

**Correct. CMMI isn't a valid Project Boards template. The templates that can be automated and already configured are: Basic kanban, Automated kanban, Automated kanban with review, and Bug triage.**

Bottom of Form

**Next unit: Summary**

**Summary**

Completed100 XP

* 3 minutes

This module introduced you to GitHub Projects, GitHub Project Boards, and Azure Boards. It explored ways to link Azure Boards and GitHub, configure GitHub Projects and Project views, and manage work with GitHub Projects.

You learned how to describe the benefits and usage of:

* Describe GitHub Projects and Azure Boards.
* Link Azure Boards and GitHub.
* Configure and Manage GitHub Projects and boards.
* Customize Project views.

**Learn more**

* [Quickstart for projects (beta) - GitHub Docs](https://docs.github.com/issues/trying-out-the-new-projects-experience/quickstart).
* [About project boards - GitHub Docs](https://docs.github.com/issues/organizing-your-work-with-project-boards/managing-project-boards/about-project-boards).
* [What is Azure Boards? Tools to manage software development projects. - Azure Boards | Microsoft Learn](https://learn.microsoft.com/en-us/azure/devops/boards/get-started/what-is-azure-boards).
* [Azure Boards-GitHub integration - Azure Boards | Microsoft Learn](https://learn.microsoft.com/en-us/azure/devops/boards/github).
* [Managing iterations in projects (beta) - GitHub Docs](https://docs.github.com/issues/trying-out-the-new-projects-experience/managing-iterations).
* [Quickstart for projects (beta) - GitHub Docs](https://docs.github.com/issues/trying-out-the-new-projects-experience/quickstart).
* [Best practices for managing projects (beta) - GitHub Docs](https://docs.github.com/issues/trying-out-the-new-projects-experience/best-practices-for-managing-projects).
* [Customizing your project (beta) views - GitHub Docs](https://docs.github.com/issues/trying-out-the-new-projects-experience/customizing-your-project-views).
* [About team discussions - GitHub Docs](https://docs.github.com/organizations/collaborating-with-your-team/about-team-discussions).

**All units complete:**

“” <https://learn.microsoft.com/en-us/training/modules/plan-agile-github-projects-azure-boards/14-summary#completion> “”