# Repository Link

<https://github.com/Tom-J991/Task3-Repository>A screenshot of a computer

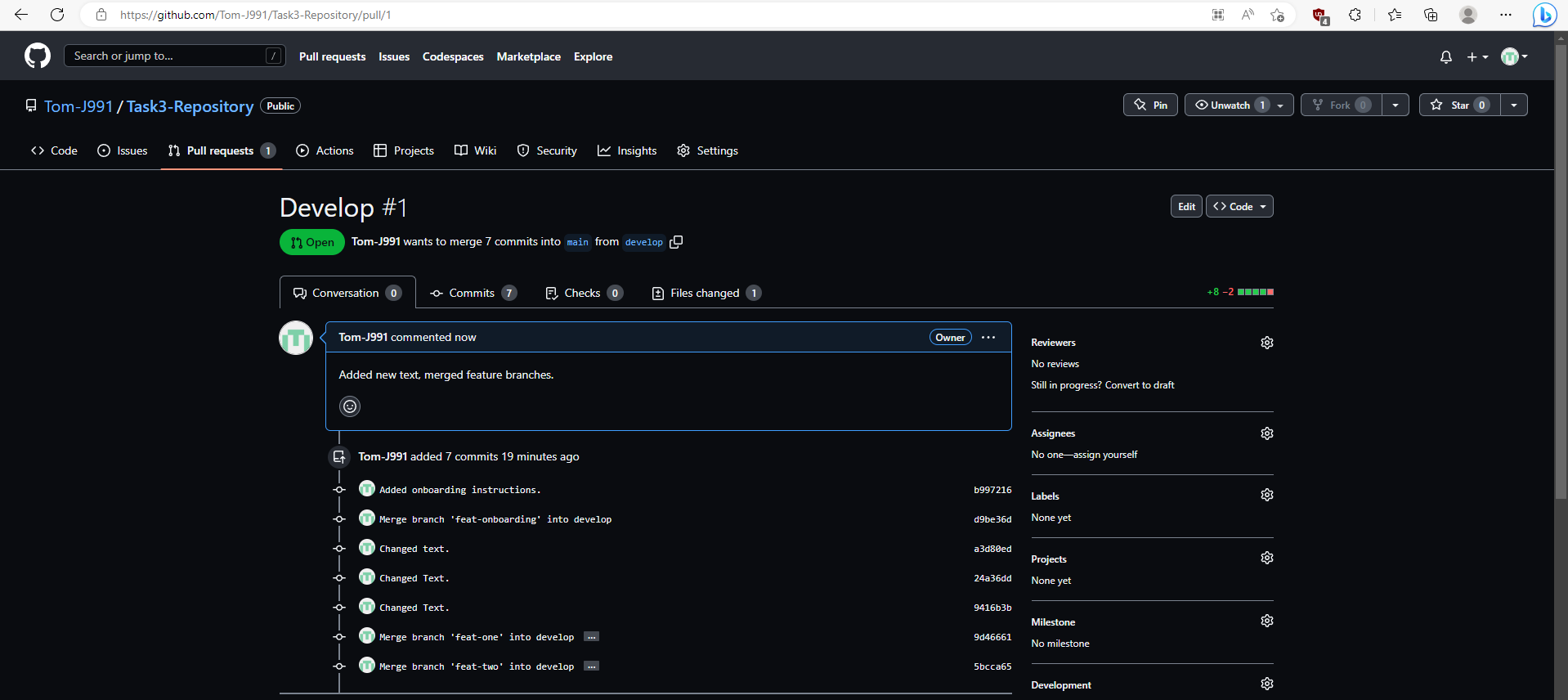
Description automatically generated

# Merge Commit

<https://github.com/Tom-J991/Task3-Repository/commit/d9be36d2509dfec907b736fba81c2774497261a7>Graphical user interface, application

Description automatically generated

# Pull Request

<https://github.com/Tom-J991/Task3-Repository/pull/1>

# Problem Statement

When working on large projects a developer may face a number of problems when making and keeping track of changes to the projects codebase as it becomes harder to merge changes between multiple copies of the project and keep track of backups and updates, these are problems faced by nearly every developer, team, and studios.

Developers need a solution that allows them to easily keep track of, merge, and undo changes between files, create backups, and keep a database of the project, while allowing easy collaboration between multiple developers. A solution like this is offered by a system known as Version Control. Multiple software solutions exist for version control, for example:

* Perforce,
* Mercurial,
* and SVN, also known as Subversion.

One of the most popular options for version control today is known as Git.

Git allows developers to create repositories for a project and store them either locally or on a server, a repository acts as a database for the project and has a list of changes showing the history of the repository. A big advantage with Git is that many repository providers exist to allow people to store their own git repositories in the cloud such as: GitHub, Bitbucket, and GitLab. Most of these repository providers have extra functionality such as issue tracking that allows developers to share various bugs and issues with the project.

Git uses a distributed model that allows a developer to store a full copy of the main repository locally on their machine, make changes to it, and easily push all the changes back to the host repository and have those changes merged into the main repository, this is known as a commit.   
Newer changes in the main repository can also be pulled back to a local repository so that the developer can work on newer versions of the project. This functionality is great for large teams and studios as it allows for easy collaboration.

Another big feature and advantage of Git is branching. This allows developers, teams, and studios to create copies of the repository with their own history and changes that can be fully merged with other branches such as the default main/or master branch. This is especially useful in large teams and studios as it allows work on the project to occur concurrently rather than waiting to get a latest version before continuing your work. Git has additional features that helps developers to deal with conflicts when merging branches together, Git will tell developers where a conflict has occurred when merging a file allowing the developer to fix it manually where Git cannot.

Additional functionality with Git includes - Commenting with commits so that developers can easily know what the commit has changed, and also comparing changes within different branches and commits.

The main disadvantages to using Git is that it can be considered to be difficult to learn and understand at first, some developers consider it to work slower on specific operating systems such as Windows, there is no built-in access control system, and the Git software is only a CLI (Command Line Interface), though software does exist to give Git a proper GUI (Graphical User Interface) frontend such as GitKraken, Sourcetree, and GitHub’s own desktop app.

# Resource List

1. [Create and push a branch to the remote repository (Git) | Sourcetree | Atlassian Documentation](https://confluence.atlassian.com/get-started-with-sourcetree/create-and-push-a-branch-to-the-remote-repository-git-847359118.html)
2. [Use Sourcetree branches to merge an update | Bitbucket Cloud | Atlassian Support](https://support.atlassian.com/bitbucket-cloud/docs/use-sourcetree-brances-to-merge-an-update/)
3. [About pull requests - GitHub Docs](https://docs.github.com/en/pull-requests/collaborating-with-pull-requests/proposing-changes-to-your-work-with-pull-requests/about-pull-requests)
4. [Creating a pull request - GitHub Docs](https://docs.github.com/en/pull-requests/collaborating-with-pull-requests/proposing-changes-to-your-work-with-pull-requests/creating-a-pull-request)
5. [Git - Documentation (git-scm.com)](https://git-scm.com/doc)
6. [About - Git (git-scm.com)](https://git-scm.com/about)
7. [Introduction To GIT | Guide to GIT Component, Application, Characteristics (educba.com)](https://www.educba.com/introduction-to-git/)