|  |  |
| --- | --- |
| **Git Flow** | |
| **Advantages** | **Disadvantages** |
| Defined structure with specific branches for each type of development. Facilitates the management and monitoring of project progress. | Complex structure for small teams. Git history becomes unreadable. |
| Separate master (production) and develop (development) branches help define the scope of the test. To test only specific branches. | Can be difficult to integrate with CI/CD tools (software applications designed to support continuous integration and continuous delivery/deployment practices) |
| Ideal for handling multiple versions of production code. | Not recommended when users need to maintain a single production version. |
| It facilitates collaboration as multiple developers can work on different features simultaneously without interfering with each other. | This strategy can overcomplicate the source control depending on the scope of the project. |

**Source Code Management – SCM**

Disadvantages and advantages of branching models

A diagram of a process

Description automatically generated

|  |  |
| --- | --- |
| **GitHub Flow** | |
| **Advantages** | **Disadvantages** |
| It focuses on one main branch, usually "master", and uses other branches for new features or fixes. | By not having specific branches for stable releases, it can be more difficult to maintain a stable master branch. And cause conflicts if team members work on different functionalities simultaneously. |
| By having a simpler structure, it facilitates the continuous flow of work. Clean and easily readable Git history. Ability to easily integrate with CI/CD. | It can be complicated to manage hotfixes or urgent corrections, since there is no specific structure for these situations. |
| Ideal when you need to maintain a single production version. | It is not suitable for handling multiple versions of code. |
| It adapts well to small teams, as it allows greater freedom in development. Facilitates collaboration in real time. Pull requests are prioritized. | By enabling a faster flow of changes, it can be more difficult to maintain high quality standards if there is no emphasis on review and testing before merging to the master branch. |

A diagram of a process

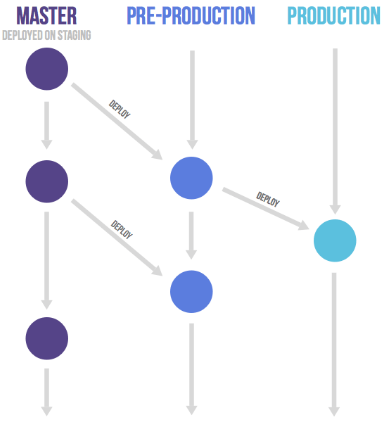
Description automatically generated

|  |  |
| --- | --- |
| **Trunk Based Development (TBD)** | |
| **Advantages** | **Disadvantages** |
| By maintaining a single master branch, you simplify code management and reduce branching complexity. Shorter feedback cycles for developers as code changes become quickly visible. | If integration and testing practices are not implemented, direct changes to the master branch can lead to further code instability. |
| True continuous integration as developers constantly keeps the trunk updated. | For significant changes that could affect stability, it can be difficult to integrate them without disrupting the main workflow. |
| Excellent choice for CI/CD pipelines with simpler workflows for automated testing. | Need a rigorous approach to code quality to ensure that changes dont negatively impact the project. |
| Working on a common branch promotes direct collaboration and code review between team members by allowing you to see each other's changes quickly while reducing code conflicts. |  |

A diagram of a software program

Description automatically generated

|  |  |
| --- | --- |
| **GitLab Flow** | |
| **Advantages** | **Disadvantages** |
| It combines feature-based development and feature branches. It includes branches such as development, pre-production and production. | It can be complex to implement with the added overhead of managing environmental branches. |
| Easily integrates into CI/CD processes. | Development branches can become complicated and confusing if not managed properly |
| Provides proper isolation between environments and ensures a clean state in the branches. Easier, cleaner to read the git history |  |
| Encourages collaboration through the use of merge requests, allowing review before merging into the master branch. |  |



<https://www.bmc.com/blogs/devops-branching-strategies>

<https://www.abtasty.com/blog/git-branching-strategies/#gitflow>