Three branching strategies for agile teams

**Branch** types  
  
There are several types of branches that are frequently used in software development. This section explains what each branch type is for, and the typical prefix convention for each branch type. In Bitbucket, the prefix can be changed for all branches other than development or production.

|  |  |
| --- | --- |
| **Development branch**  Usually the integration branch for feature work and is often the default branch or a named branch. For pull request workflows, the branch where new feature branches are targeted. | main  or develop |
| **Production branch**  Used for deploying a release. Branches from, and merges back into, the development branch. In a Git-based workflow it is used to prepare for a new production release. | varies |
| **Feature branch**  Used for specific feature work or improvements. Generally branches from, and merges back into, the development branch, using pull requests. | feature/ |
| **Release branch**  Used for release tasks and long-term maintenance versions. They are branched from the development branch and then merged into the production branch. | release/ |
| **Bugfix branch**  Typically used to fix Release branches. | bugfix/ |
| **Hotfix branch**  Used to quickly fix a Production branch without interrupting changes in the development branch. In a Git-based workflow, changes are usually merged into the production and development branches. | hotfix/ |

Azure Devops agent:  
  
Azure DevOps agent is a lightweight, cross-platform agent that runs on a machine to enable communication and data transfer between Azure DevOps and the machine. It allows developers to run build and release pipelines on their own servers or machines, instead of relying on Microsoft-hosted agents  
  
The agent can be installed on Windows, Linux, and macOS machines, and can be configured to run as a service or a standalone process. Once the agent is installed and configured, it can communicate with Azure DevOps to download source code, build and test applications, and deploy releases.

Azure custom agent vs self-hosted agents:

The main difference between custom agents and self-hosted agents is that custom agents are managed by Microsoft, while self-hosted agents are managed by the user or organization. Custom agents are ideal for teams that want a fast and simple way to build and deploy applications, while self-hosted agents are ideal for teams that require more customization and control over the build and deployment process.

Azure devops ubuntu agent:

To run build and release pipelines on Ubuntu, you can set up an Azure DevOps Ubuntu agent. Here are the general steps:

1. Set up an Ubuntu machine (either physical or virtual) that meets the system requirements for the agent.
2. Install the necessary dependencies on the Ubuntu machine, including the .NET Core runtime, Node.js, and Docker (if needed for your build/release pipeline).
3. Create a new agent pool in Azure DevOps and add your Ubuntu machine to that pool.
4. Download and install the Azure DevOps agent on your Ubuntu machine.
5. Configure the agent to use your Azure DevOps organization and project, and start the agent service.
6. Verify that the agent is connected to Azure DevOps by checking the agent pool in the Azure DevOps UI.

Once the agent is set up and connected to Azure DevOps, you can use it in your build and release pipelines to build, test, and deploy your applications on Ubuntu. You can also customize the agent configuration to meet your specific needs, such as installing additional dependencies or setting environment variables.

GCP Cloud run :

GCP Cloud Run is a serverless platform that allows developers to deploy and run containerized applications without worrying about the underlying infrastructure. When deploying applications to Cloud Run, there are a few networking considerations to keep in mind.

1. Access control: Cloud Run services can be accessed publicly or privately. By default, Cloud Run services are accessible over the internet. However, you can configure private access using VPC (Virtual Private Cloud) connectors to restrict access to your service.
2. Load balancing: Cloud Run services are automatically load balanced by Google's infrastructure. When you deploy a Cloud Run service, it is automatically scaled up or down based on traffic demand, and the traffic is distributed across multiple instances to ensure availability and performance.
3. Custom domains: You can map custom domains to your Cloud Run service, which allows you to use your own domain name instead of the default **\*.run.app** URL. This can improve branding and make it easier for users to access your service.
4. IAM roles: Cloud Run uses IAM (Identity and Access Management) roles to manage access to your service. You can assign IAM roles to users and service accounts to control who has access to your Cloud Run service.
5. Monitoring: Cloud Run integrates with Cloud Monitoring, which allows you to monitor the performance and availability of your service. You can set up alerts and dashboards to track metrics such as request latency, error rate, and more.

Overall, GCP Cloud Run provides a highly scalable and flexible platform for deploying containerized applications, with robust networking features that allow developers to customize and secure their services as needed.

1. Access control: Cloud Run services can be accessed publicly or privately. By default, Cloud Run services are accessible over the internet. However, you can configure private access using VPC (Virtual Private Cloud) connectors to restrict access to your service.

2. Load balancing: Cloud Run services are automatically load balanced by Google's infrastructure. When you deploy a Cloud Run service, it is automatically scaled up or down based on traffic demand, and the traffic is distributed across multiple instances to ensure availability and performance.

3. Custom domains: You can map custom domains to your Cloud Run service, which allows you to use your own domain name instead of the default \*.run.app URL. This can improve branding and make it easier for users to access your service.

4. IAM roles: Cloud Run uses IAM (Identity and Access Management) roles to manage access to your service. You can assign IAM roles to users and service accounts to control who has access to your Cloud Run service.

Monitoring: Cloud Run integrates with Cloud Monitoring, which allows you to monitor the performance and availability of your service. You can set up alerts and dashboards to track metrics such as request latency, error rate, and more