**ARTH - Task 29 👨🏻‍💻  
Task Description📄  
✍🏻 Research for industry use cases of Jenkins**

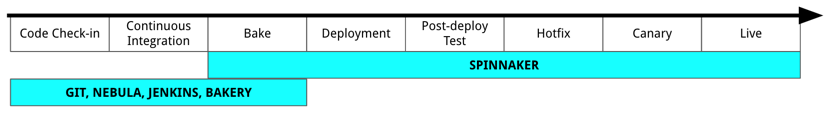
****





How the Code at Netflix is Build!!!

**How does Netflix build code before it’s deployed to the cloud? Here is the tools and techniques used to go from source code to a deployed service serving movies and TV shows to more than 75 million global Netflix members.**



**There are a number of steps that need to happen before a line of code makes it way into Spinnaker:**

**Code is built and tested locally using Nebula**

**Changes are committed to a central git repository**

**A Jenkins job executes Nebula, which builds, tests, and packages the application for deployment**

**Builds are “baked” into Amazon Machine Images**

**Spinnaker pipelines are used to deploy and promote the code change**

**…………………………………………………………………**

**Once a line of code has been built and tested locally using Nebula, it is ready for continuous integration and deployment. The first step is to push the updated source code to a git repository. Teams are free to find a git workflow that works for them.**

**Once the change is committed, a Jenkins job is triggered. Our use of Jenkins for continuous integration has evolved over the years. We started with a single massive Jenkins master in our datacenter and have evolved to running 25 Jenkins masters in AWS. Jenkins is used throughout Netflix for a variety of automation tasks above just simple continuous integration.**

**A Jenkins job is configured to invoke Nebula to build, test and package the application code. If the repository being built is a library, Nebula will publish the .jar to our artifact repository. If the repository is an application, then the**[**Nebula ospackage plugin**](https://github.com/nebula-plugins/nebula-ospackage-plugin)[**will be executed**](https://github.com/nebula-plugins/nebula-ospackage-plugin)**. Using the Nebula ospackage (short for “operating system package”) plugin, an application’s build artifact will be bundled into either a Debian or RPM package, whose contents are defined via a simple Gradle-based DSL. Nebula will then publish the Debian file to a package repository where it will be available for the next stage of the process, “baking”.**

**x-x-x-x-x-x-x-x-x-x-x-x-x-x-x-x**