**AZURE DEVOPS BUILD PIPELINE**

**WHAT IS CI/CD?**

**CI (Continuous Integration):**

CI process is an automation process that keeps building, testing, feeding back new code written/ changed by one of the developers to the team. CI process can be done by using automation tools.

Advantages:

* Reduce merge conflict
* Testable build

**CD process (Continuous Delivery):**

CD is process which automatically change new codes into a software, test the software with automation testing then have other tests before being approved then become a product for the end-users.

**Deployment pool:**

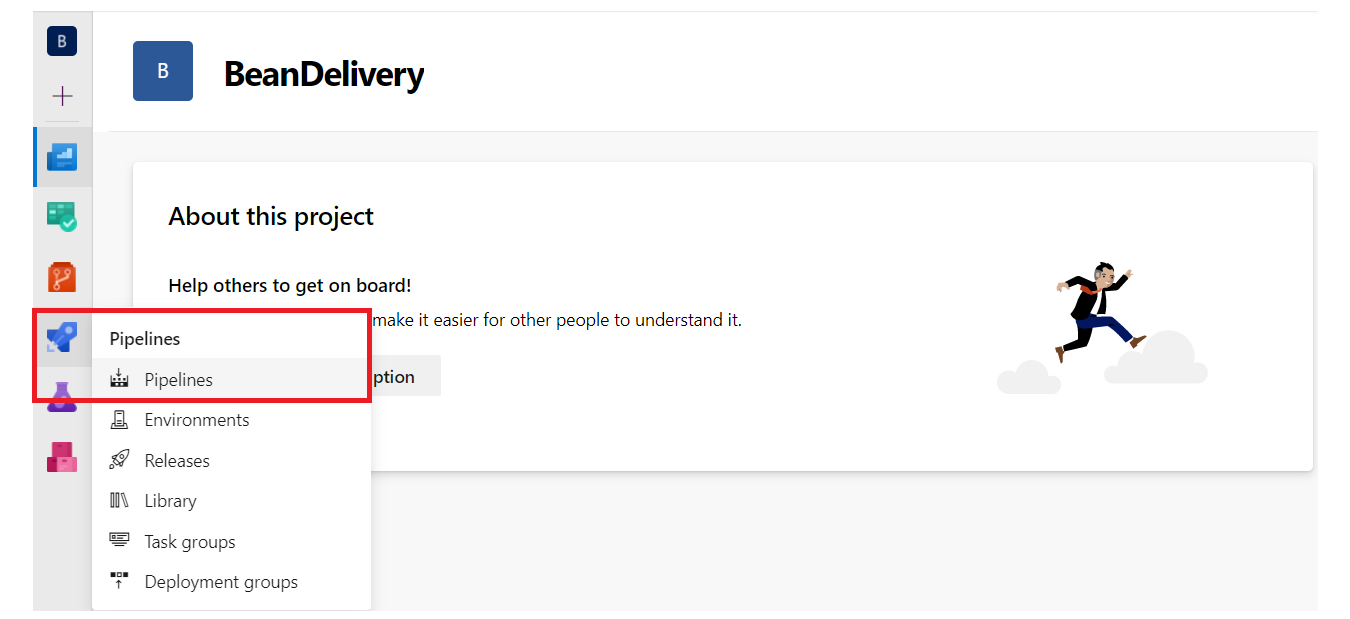
A deployment pool is a set of target servers available to the organization. When you create a new deployment pool for projects in your organization, a corresponding deployment group is automatically provisioned for each project.

**Agent pool:**

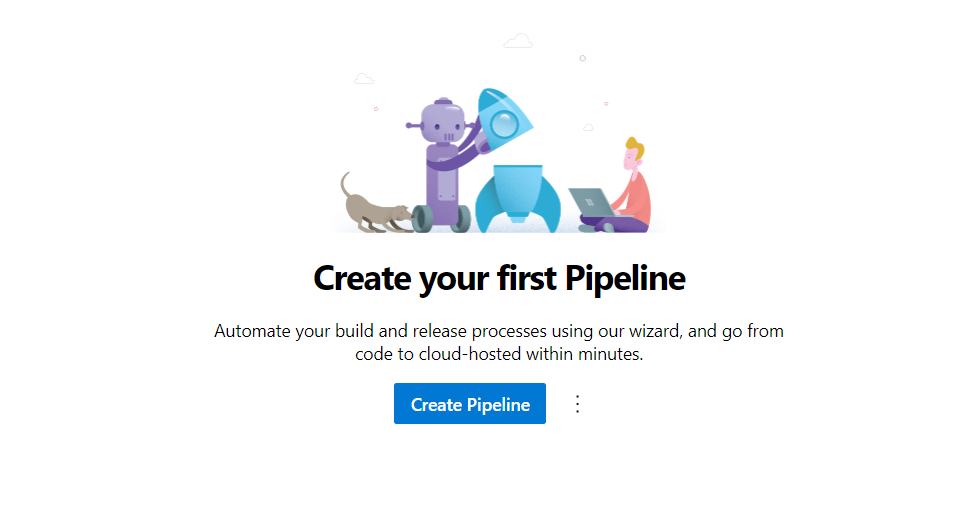
A set of services that runs the jobs defined in the pipeline (self-hosted agent, Microsoft-hosted agents).

**Build CI Pipeline with React**

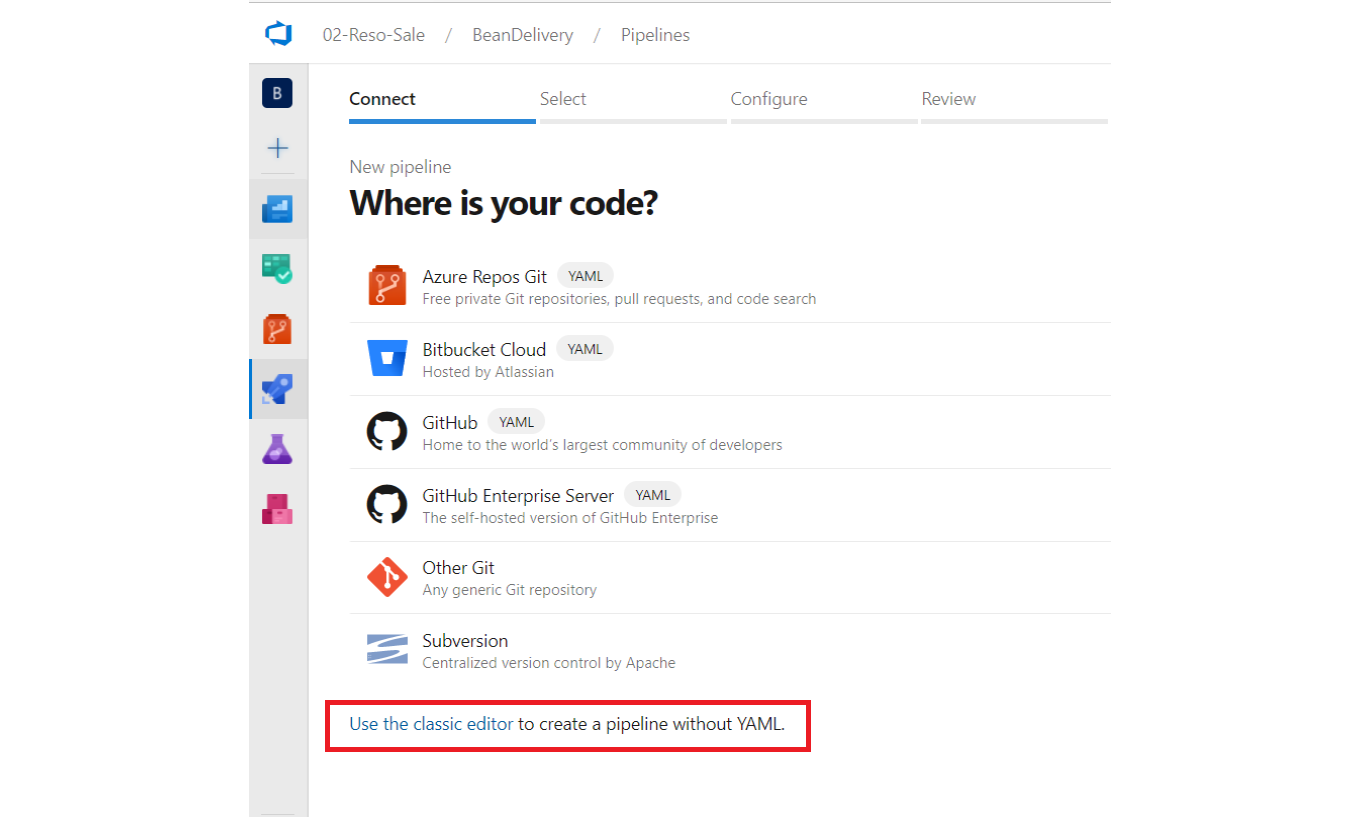
1. **Click on Pipeline section**

****

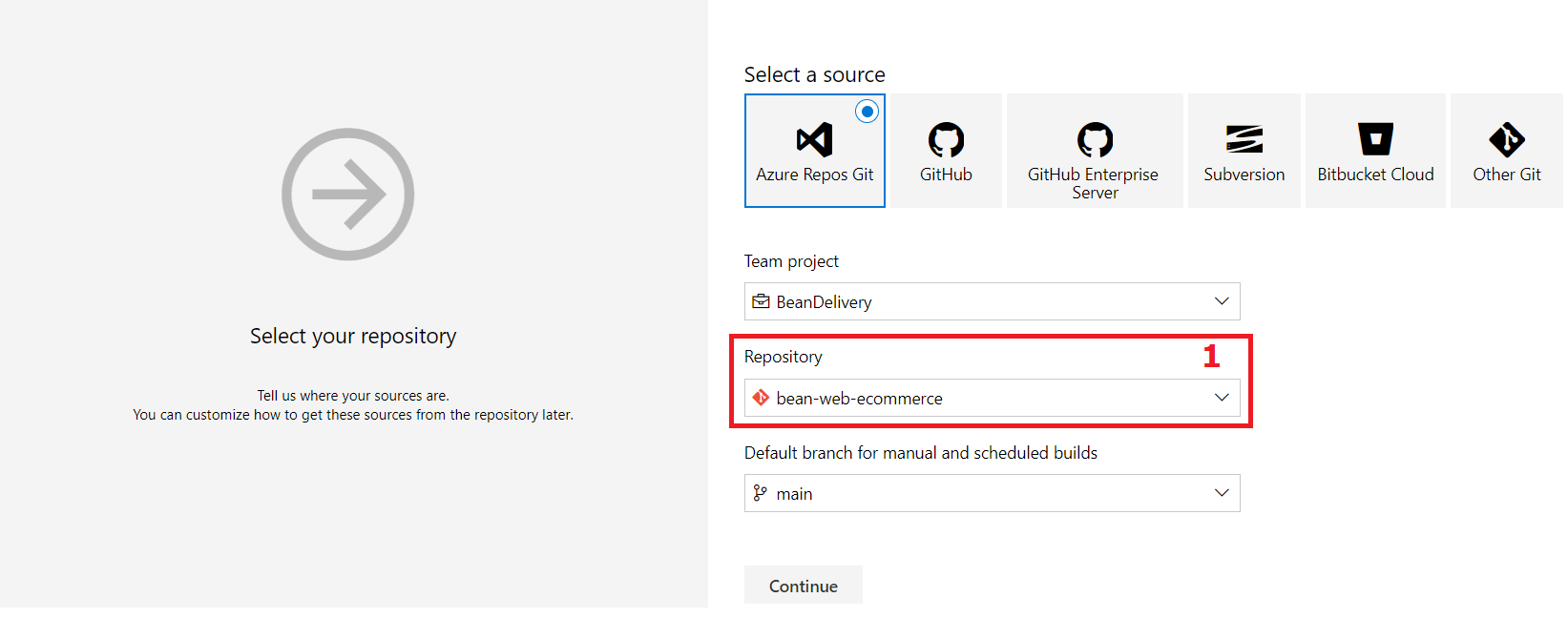
**and click create pipeline button to create a new pipeline**

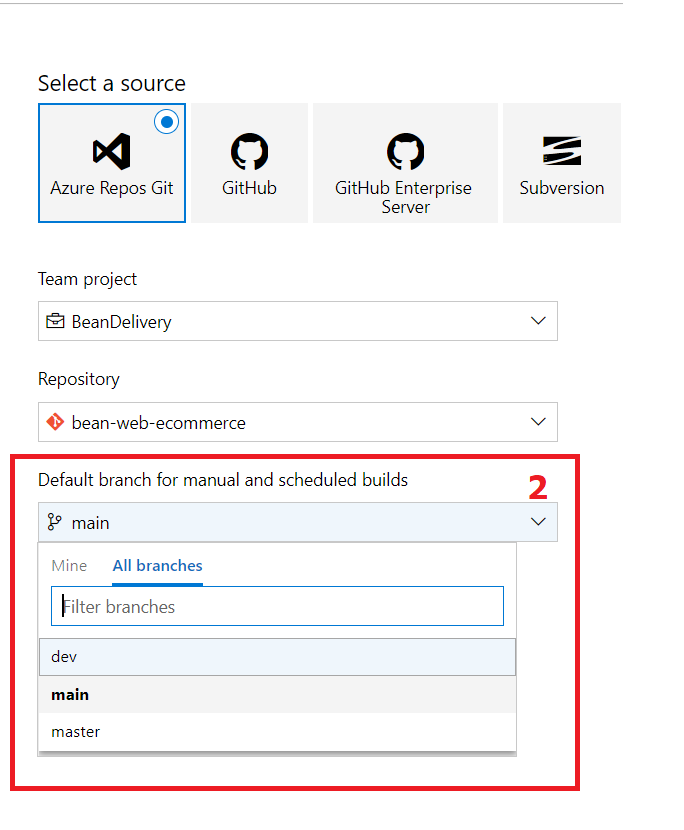
****

**Then,**

****

1. **After that, you select your repository which need to be built in pipeline**

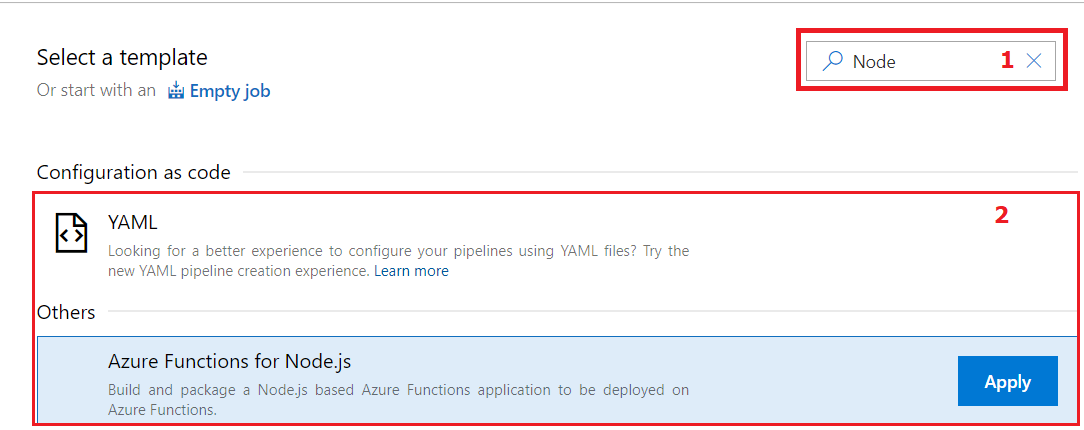
****

****

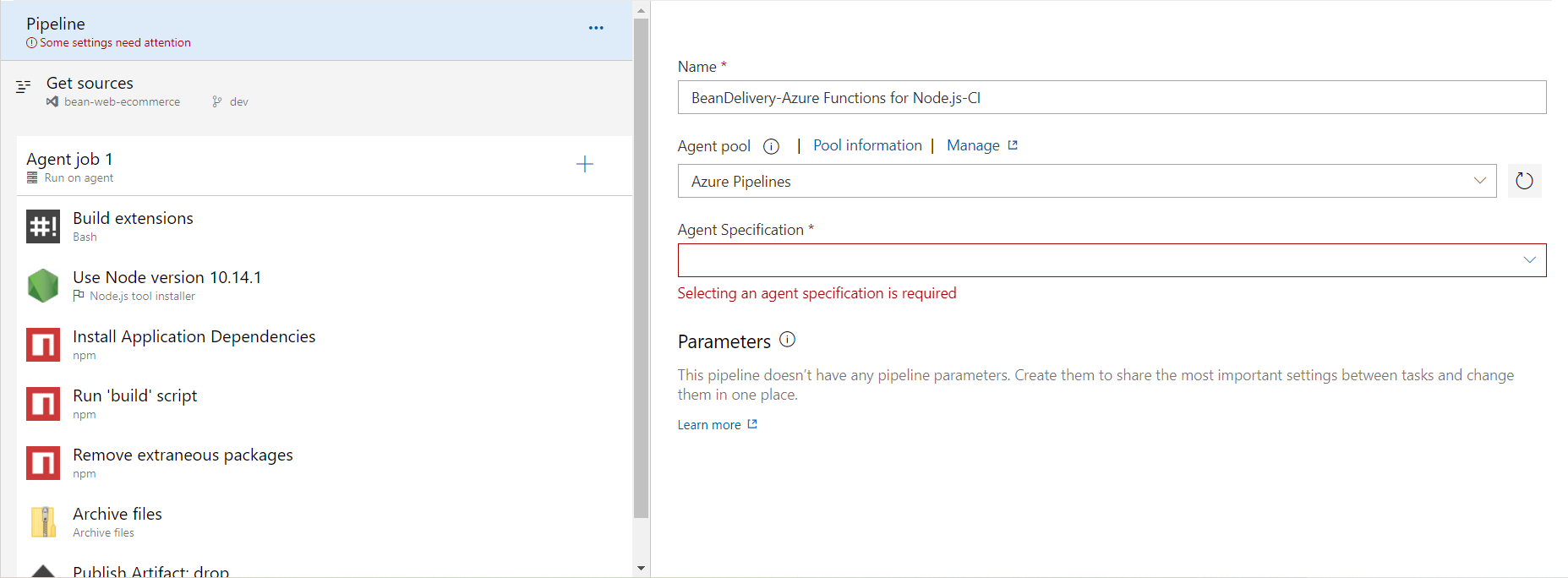
* Depend on environment pick a right branch.

1. **Then you have configured the pipeline. We have the react app so choose Node.JS with React**

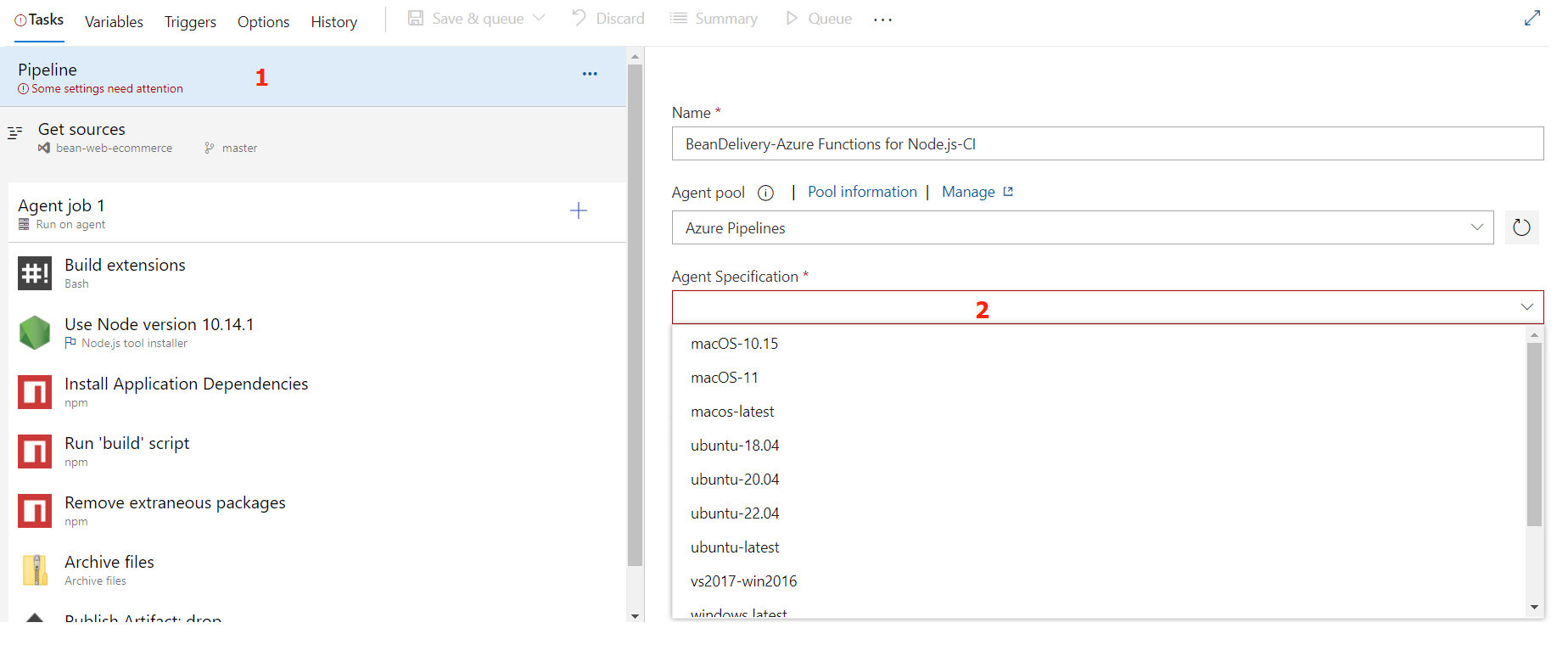
* Search with keyword “*Node*” to use the template was created by Azure

****

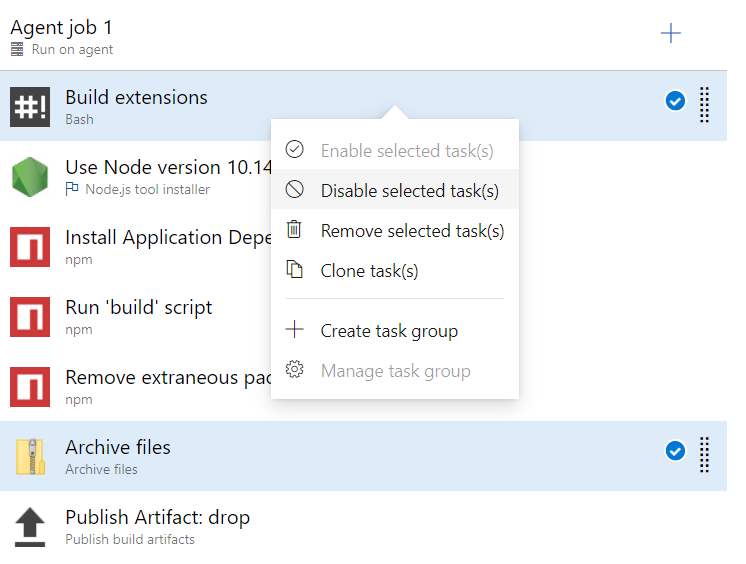
* Using the template that was created by Azure to shorten the configuration time. It has most used tasks that we have to implement into our pipeline.

****

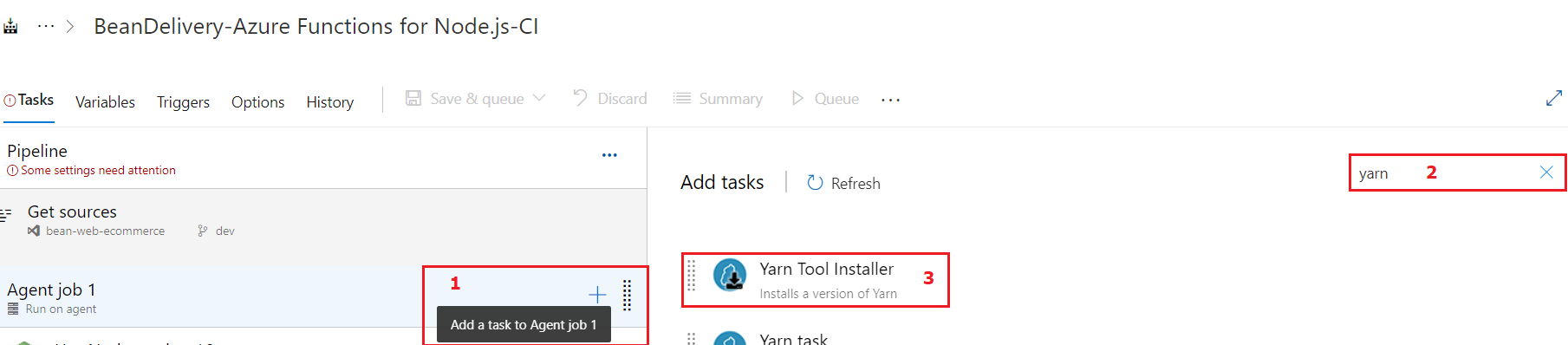
First, notice that we need to select *Agent Specification. (this Agent will be talked in another Article)*

****

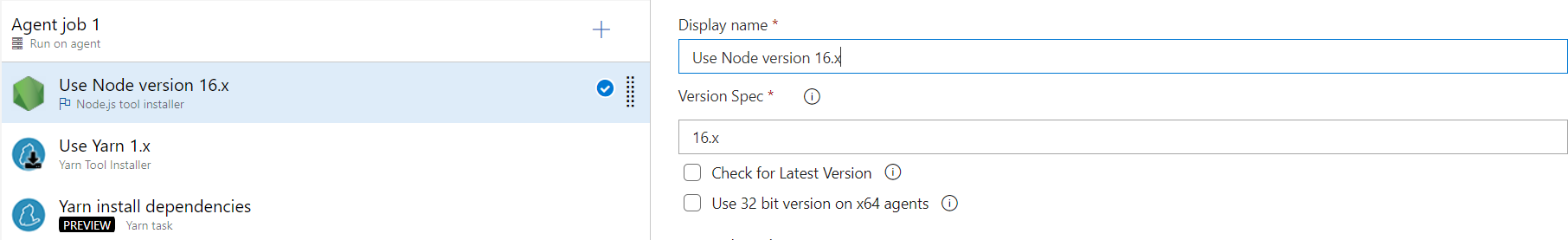
After removing/disable, I recommend using *Yarn* to execute the installation and building Node JS project (you’re able to use *npm*, if you want to).



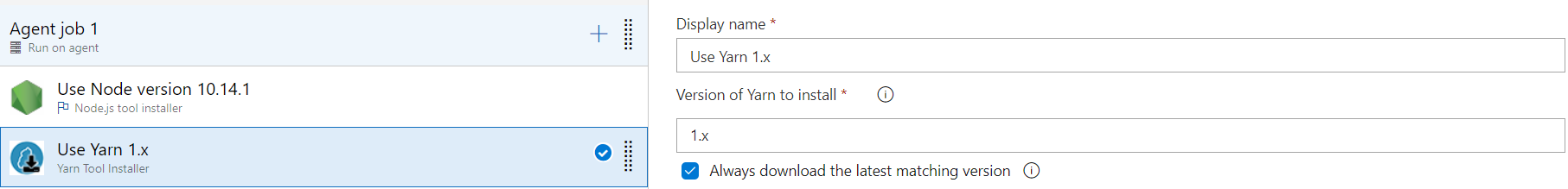
By adding these *Yarn* task, we continue to configure the important values.

****

1. Node JS version: discuss this version to Team Dev. I suggest using Node JS version 16. This is currently a stable version of Node JS.

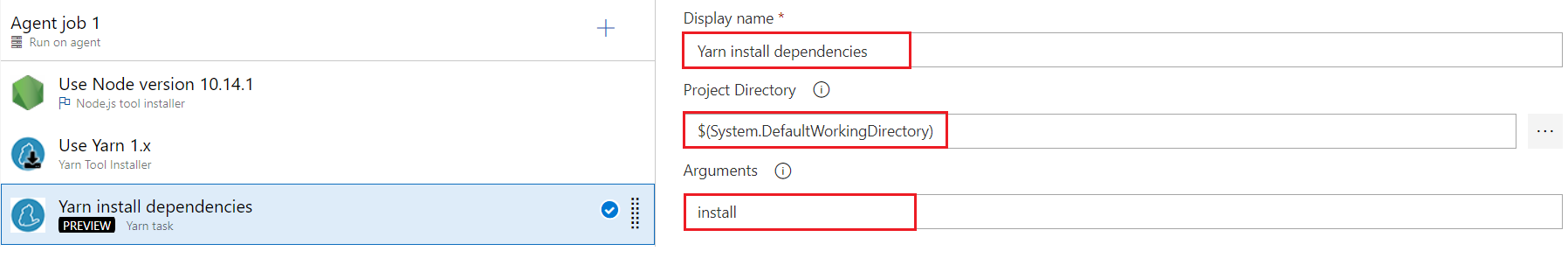


1. Use *Yarn* Task: using the latest version of it.



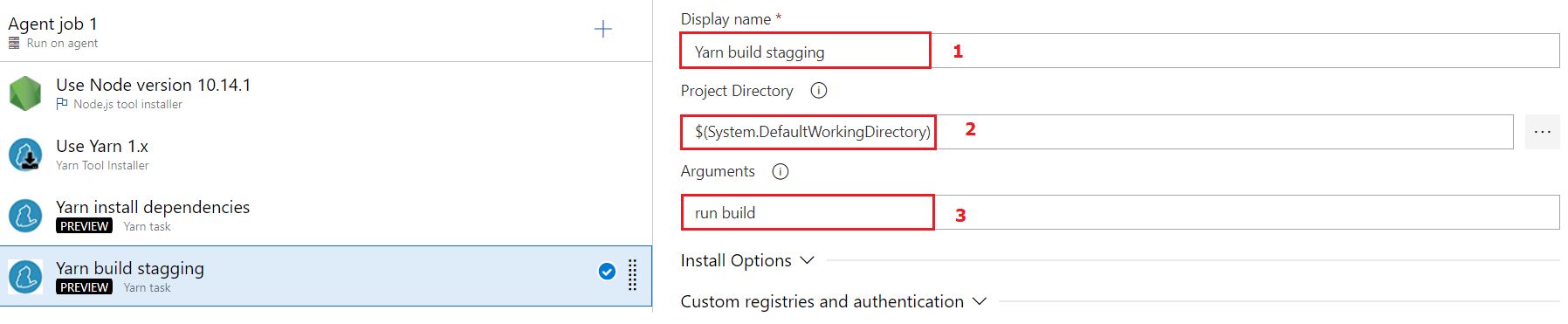
1. *Yarn* installation dependencies: this task requires you to locate the *package.json* file so that, the field *Project Directory* it should be the exact path to the file I early mentioned.

Use command “install” to fill in the *Argument* section. This means you use the command “*yarn install*” in order to install all dependencies were used in React Web App.

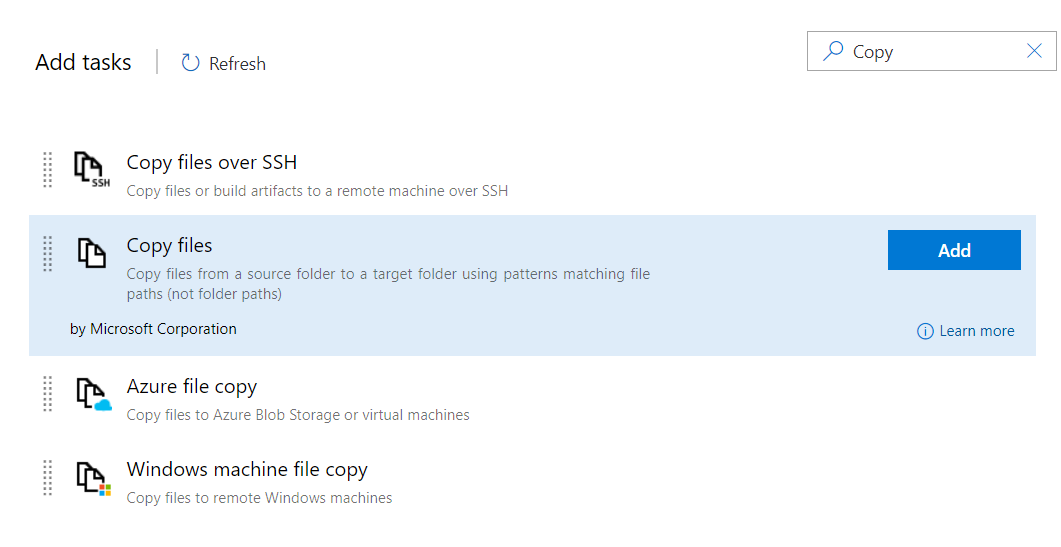


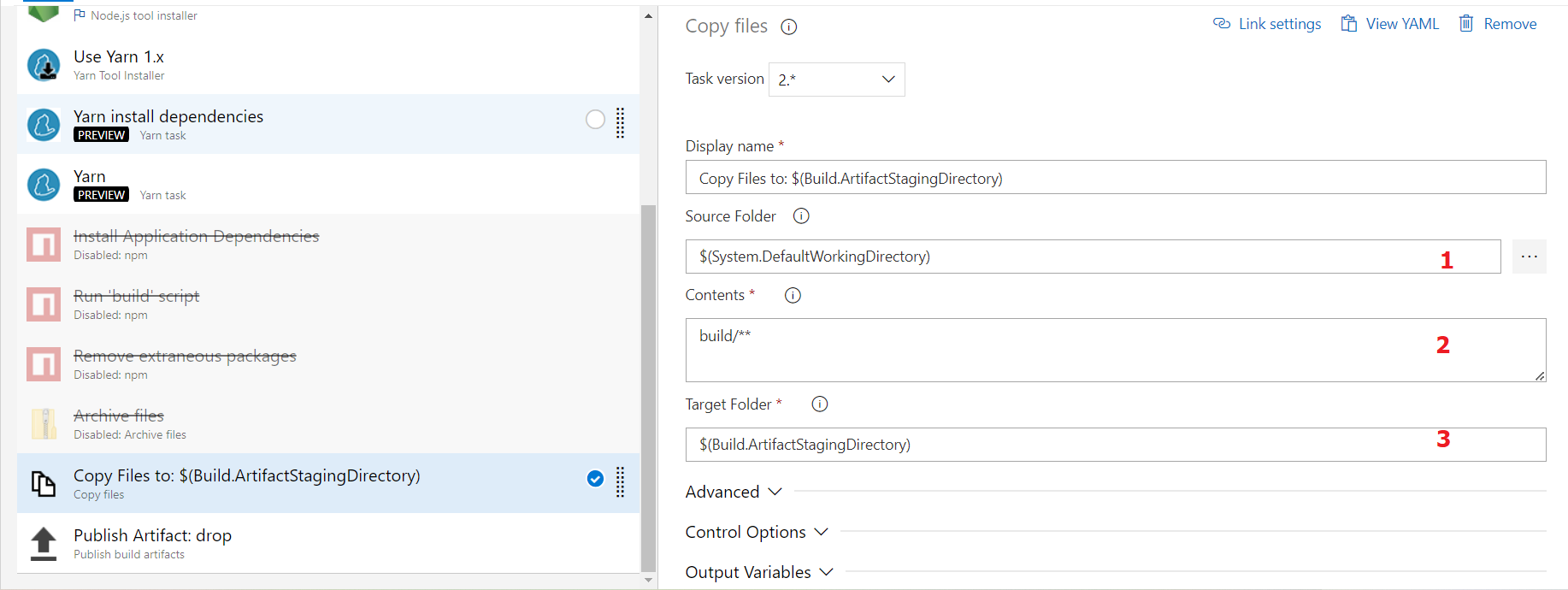
1. *Yarn* build {env} – it might be staging or production: the field *Project Directory* the same values with the previous task.

Use command “*run build*” to fill in the *Argument* section. This means you use the command “*yarn run build*” in order to build React Web App with env.staging file.

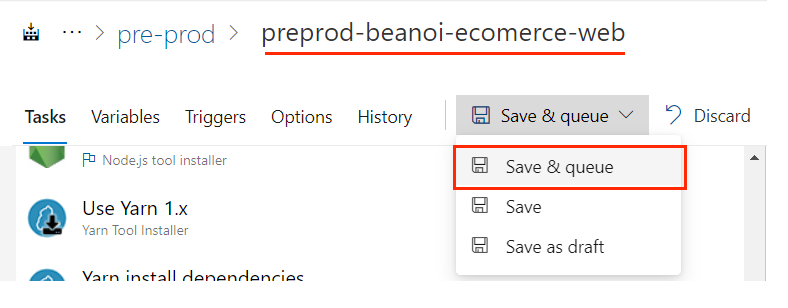


1. Next, add *Copy files* task for copying all the contents inside the *build* (it could be different to yours in this docs) folder to the publish artifact





Then save it.



Notice that, the name of pipeline always following the format:

*{env}-{project-name}-{repo-name}-CI-pipeline*

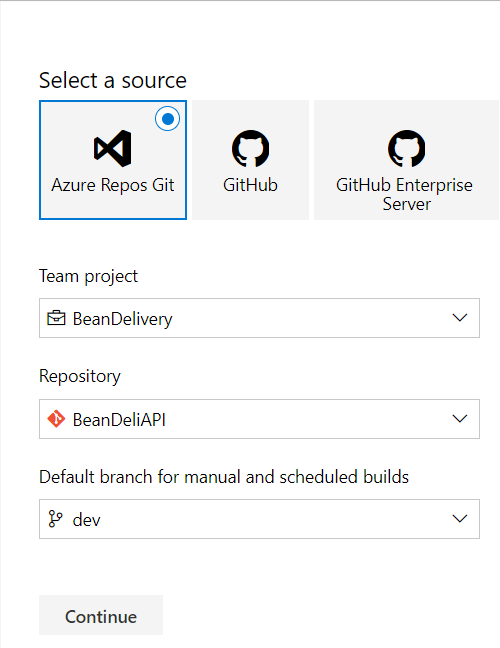
Then, you just need to wait for the result of the CI pipeline and fix bug (**if have**).

**SO, THAT’S ALL FOR BUILDING CI PIPELINE FOR NODE JS (REACT JS WEB APP)**

**Build CI Pipeline with .NET Core**

Repeat Step1 & Step2 from above

After choosing the Repo and the specific branch to build pipeline



Continue to pick a build template for **.NET Core** project

