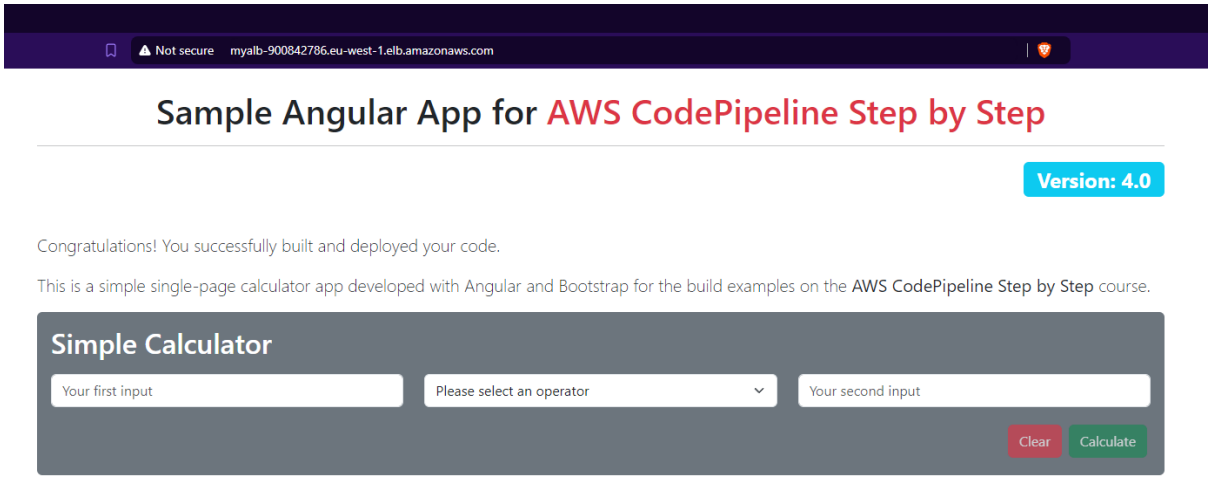
😊 In-place All at Once Deployments

This task explains the process and implications of using the "In-place All at Once" deployment method in AWS. This method deploys updates to all instances simultaneously, making it a fast deployment option. In the example, a new version (5.0) of a web application is released by pushing changes to a repository, triggering the pipeline, and deploying the updates.

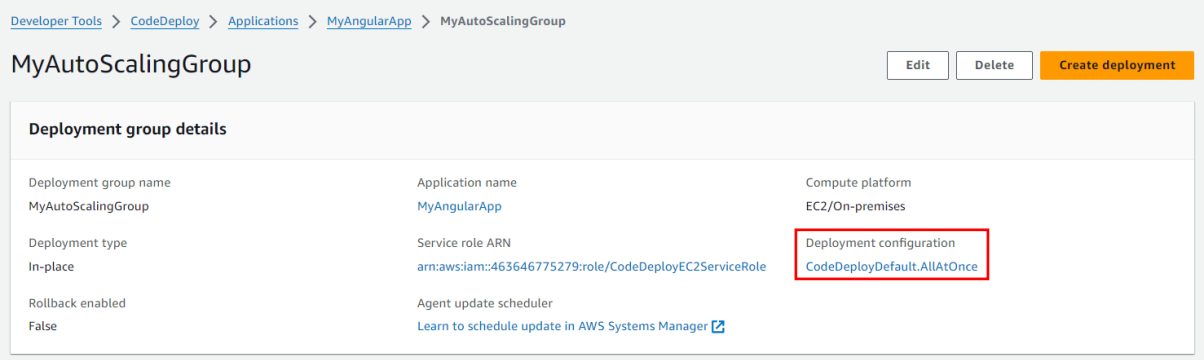
However, deploying all at once can cause temporary downtime, as all instances are updated simultaneously, which may not be suitable for production environments. Additionally, if there's an error in the deployment, it affects all instances at once. Despite these risks, this method is useful when you need to apply urgent fixes quickly. The goal is to understand the trade-offs of fast, but potentially disruptive, deployments.

😄 To begin with the Lab:

1. In the previous lab, we created a deployment group for our new auto-scaling group, updated our deploy action with it, and triggered our pipeline. We also have a load balancer routing traffic to the instances launched by this auto-scaling group. As you see, our web application is online at the DNS name of this load balancer and its version is 4.0. Now, while configuring the deployment group we did not change the default deployment type and configuration. The default deployment type is 'In-place' which means that the deployments will be performed on the existing instances.

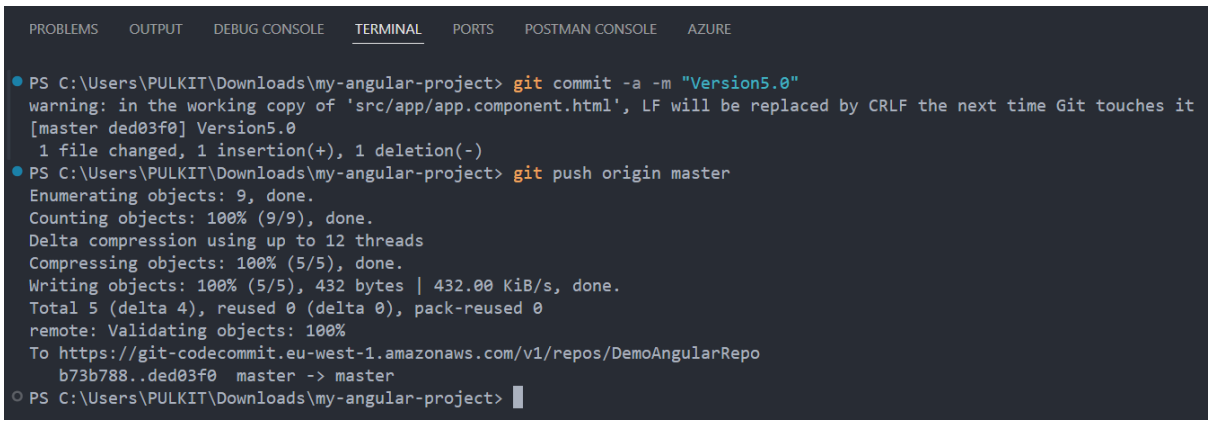


2. And the default deployment configuration is 'All At Once' which performs the deployments on all instances at the same time. This deployment method is fast but has some problems. So, in this lab, we will release a new version of our application and talk about what happens during in-place all-at-once deployments.

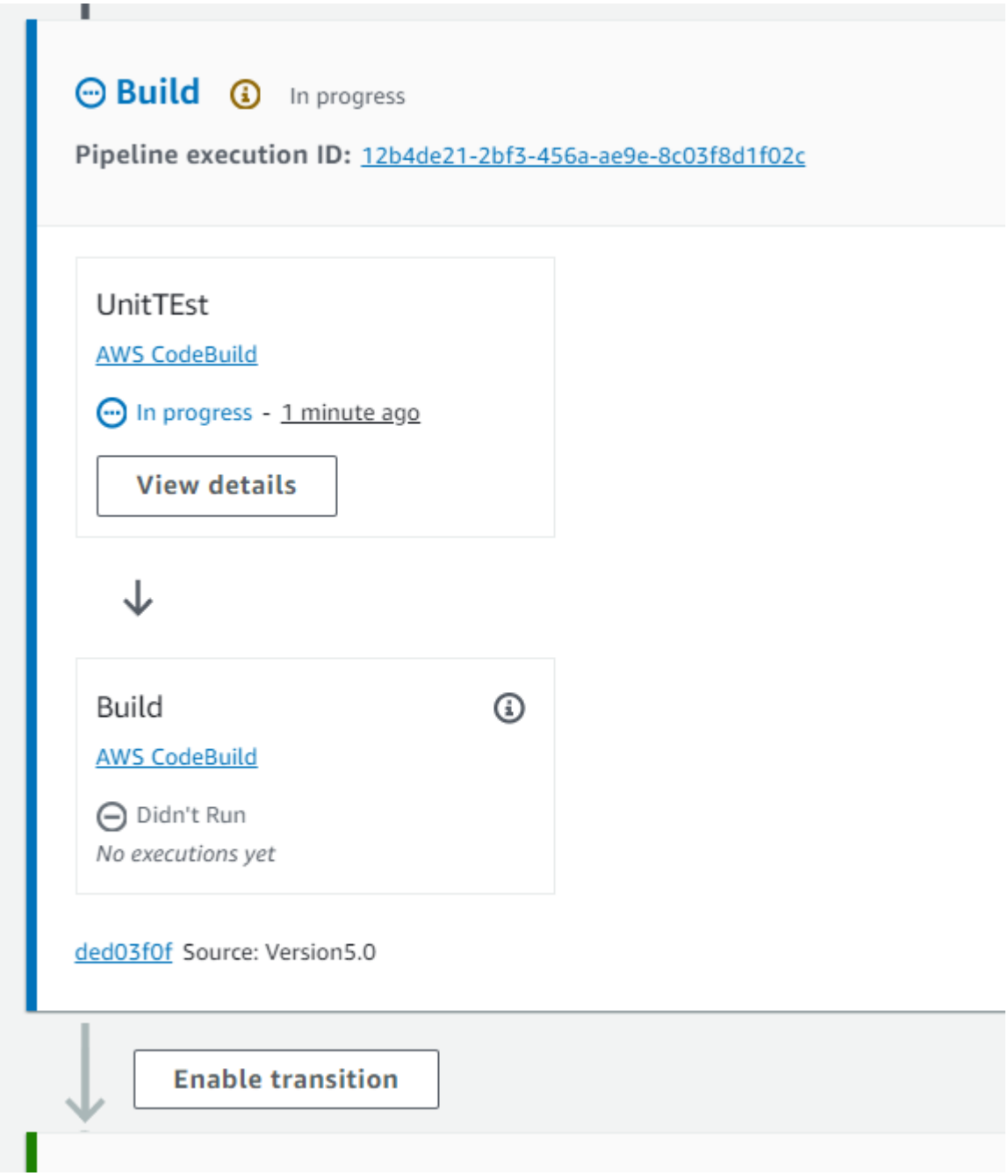


3. Now you need to open the Angular Project in VS Code and make some changes. Here you need to go to the app.component.html file and change the version of our website to 5.0, then we will open the terminal and commit changes, after that we will push the changes to our repository.

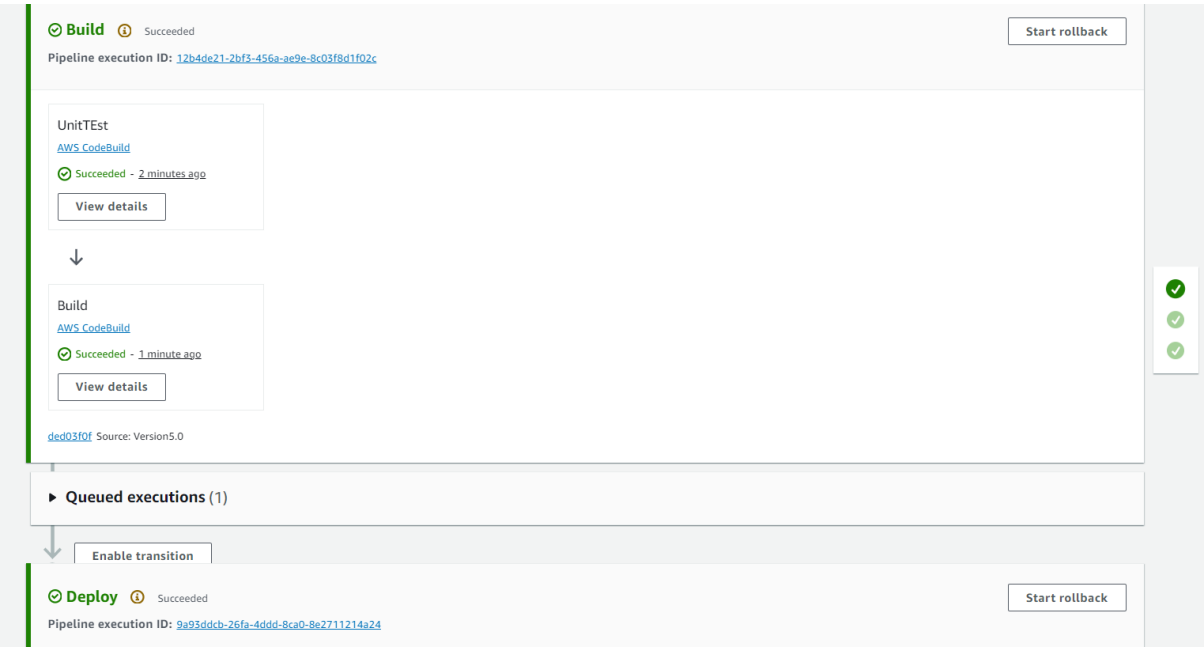
4. Below you can see that we have committed and pushed our changes to the repository. Soon after that our pipeline will start executing.



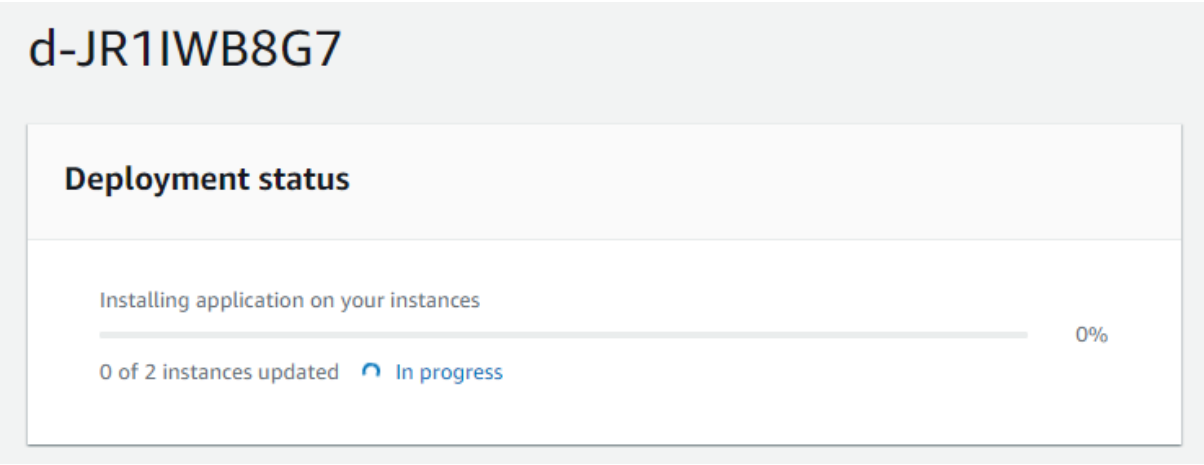
5. So, now you see that the build stage has started to get executed then at the same time you need to disable the transition between the build stage and deploy stage as you can see below.

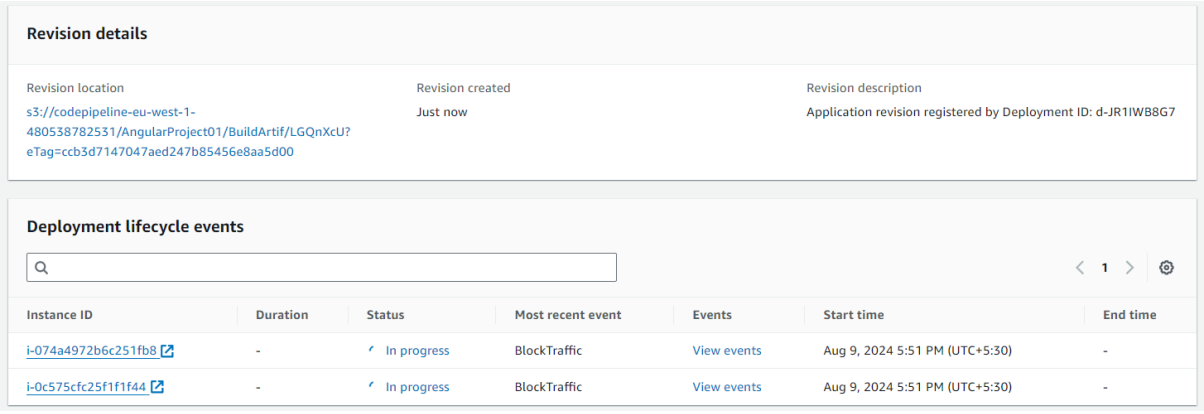


6. Below you can see that our build has been executed successfully but you will see that for the deploy stage, our execution has been queued.
7. So, now we will enable the execution, and we will wait until our deploy stage gets the option to view details.

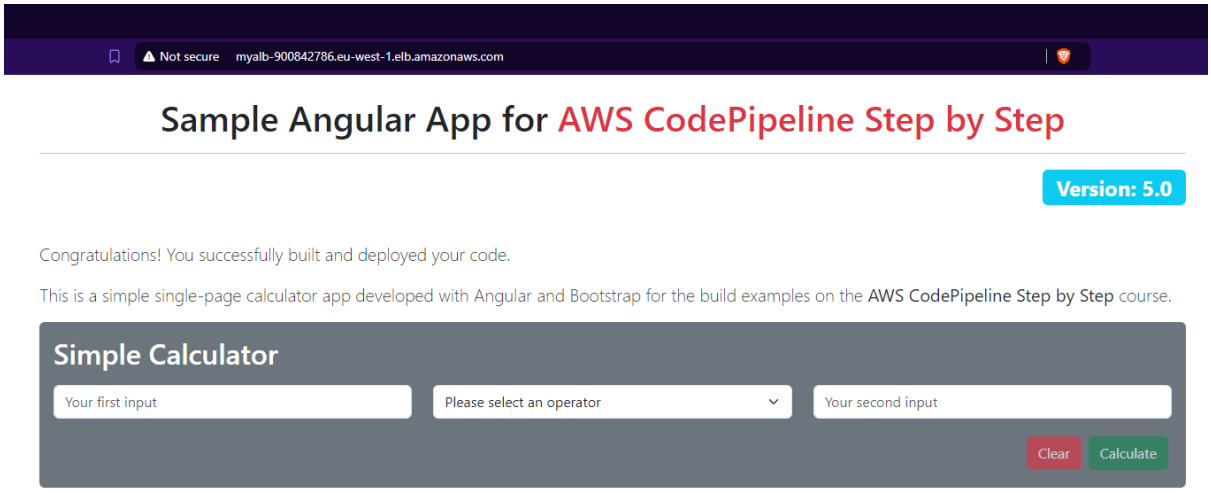


8. Below you can see that the deployment status shows us that the application is being installed on our instance. Also, if you scroll down then you will see that the deployment lifecycle events are in progress with the instances.





9. If you want to do the whole process then you need to be quick to look at most of the things which are your target group, and the auto-scaling, and then in the end refresh your website page and you will see that your website is down for some time.



10. As you see, in-place all-at-once deployments are fast because you deploy to all instances in parallel. But they are also not safe. You lose all your instances for a few minutes during the deployment which would not be acceptable for production-level applications serving traffic. Besides, if your deployment has an error it will fail in all instances simultaneously. However, this deployment method has also its use cases. For example, if you are putting out a fire on your system and need to deploy your fixes to all your instances as fast as possible.