DevOps

This lesson introduces DevOps.

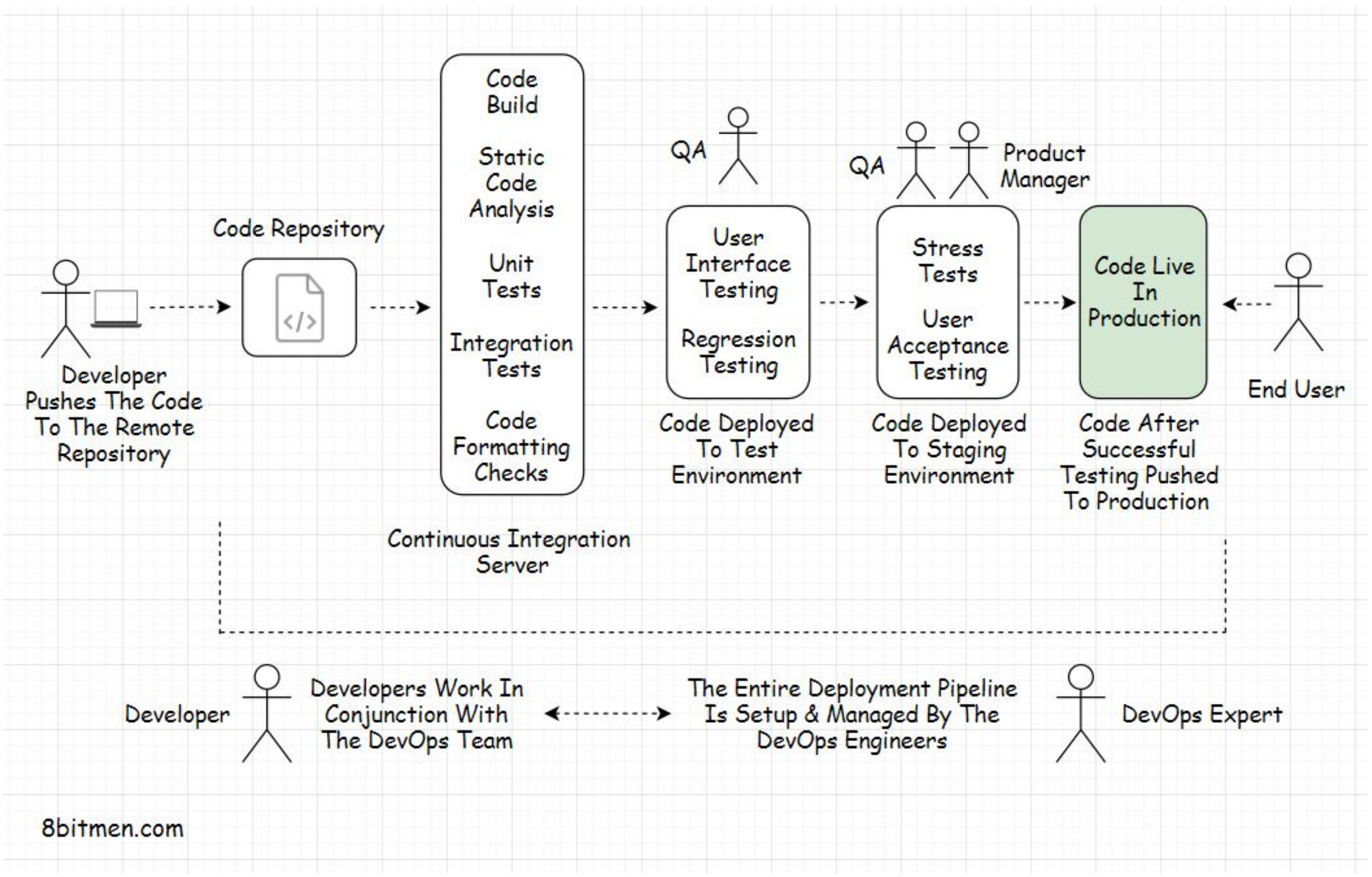
We'll cover the following

What is DevOps?

DevOps stands for *Developer Operations* which is *Development + Operations*. We are aware of the role of the developers, and *operations* entail everything right from the point the code is pushed to the remote repository to the point the live application is monitored in production.

Operations include things like:

- Setting up, managing, and monitoring the deployment pipeline; setting up the build server and end-to-end automation for all the builds and deployments
- Setting up different deployment environments such as *Dev, Staging*, and *Production*; provisioning the infrastructure and managing the *IaC*
- Monitoring different components of the system with the right tools
- Setting up and scaling application storage, diagnosing system issues, and disaster recovery
- Practicing *continuous delivery* and *deployment* in the project and adhering to the organization's processes, guidelines, etc.



Developer Pushes The Code To The Remote Repository
Code Repository
Code Build
Static Code Analysis
Unit Tests
Integration Tests
Code Formatting Checks
Continuous Integration Server
QA
User Interface Testing
Regression Testing
Code Deployed To Test Environment
QA
Product Manager
Stress Tests
User Acceptance Testing
Code Deployed To Staging Environment
Code Live In Production
Code After Successful Testing Pushed To Production
End User
Developer
Developers Work In Conjunction With The DevOps Team
The Entire Deployment Pipeline Is Setup & Managed By The DevOps Engineers
DevOps Expert
8bitmen.com

The concept of *DevOps* originates from *Agile software development methodology*. *Agile* methodology facilitates early feedback from the stakeholders or customers on the software that is being built by organizing and managing the development process into sprints.

In a sprint, which generally spans two weeks, developers write code, build new features, and showcase it to the customer for feedback.

This approach enables teams to develop software quickly and in a more reliable way, fulfilling the expectations of the customer.

The *DevOps* methodology tries to achieve the same results by facilitating better collaboration between the development and the operations team.

The *DevOps* people are in continual contact with the dev team during the whole development and the application deployment process. This enables both teams to fix issues, like system bottlenecks, come up with the right strategy to scale the system, and more. With *DevOps*, the dev and the operations teams work in collaboration as opposed to working in isolation. This approach helps ship software fast.

Before *DevOps*, in order to deploy code to production, we had to raise a ticket that

Before *DevOps*, in order to deploy code to production, we had to raise a ticket that was handled by a dedicated team set up to do that particular task. With *DevOps*,

instead of creating tickets, teams prefer to continually interact with each other using workplace communication tools like *Slack*. This smoothens the development process reducing the time it takes to deploy the code.

There is another term for *DevOps* that floats around in the industry known as *SRE (Site Reliability Engineering)*. The term was coined at *Google*, where software engineers handle the operations, ensuring the smooth functioning of all the services running at *Google*. The role of *SRE* engineers is pretty much the same as the *DevOps*.

Expectations from a DevOps engineer

We've discussed the different stages of a deployment pipeline earlier. To review, that are *coding, building the code, running tests, packing the application, releasing the code, configuration management*, and *production monitoring*.

As a *DevOps* engineer, we are expected to have an understanding of all these stages of the deployment pipeline. We should be aware of what technologies and tools should be leveraged to build a deployment pipeline, what build server to pick for a certain use case, what code management system would fit best with respect to the organization's requirements, and so on.

Based on the resources that the business has, it can have a dedicated *DevOps* team or have the developers perform the *DevOps* related tasks. This generally happens if the team size is small, like in a startup.

I've done a bit of *DevOps* work as a developer in the past, and I've also played the role of a *developer on support*. A *developer on support* has to work in collaboration with the *DevOps* team on a continual basis. They are expected to fix code issues, write new patches and etc. stuff based on their interactions with the *DevOps* team.

Working with the operations team helps a great deal in having an end-to-end understanding of the application flow.

Well, guys, this is pretty much it on *DevOps*. In the next chapter let's talk about cloud storage.

First, you'll take a quiz about the deployment workflow in the next lesson.