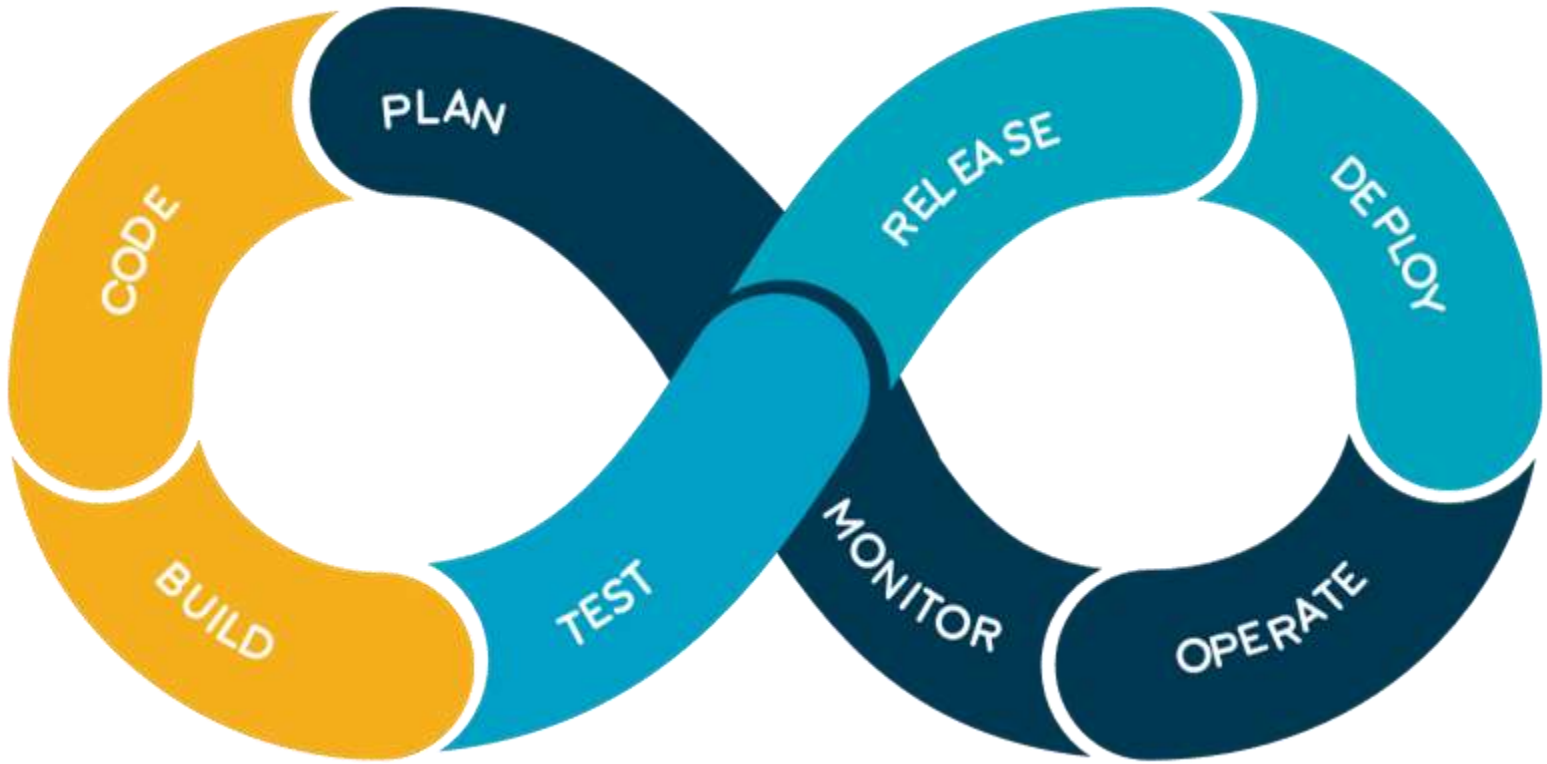
How exactly does DevOps work?

Let's move to the next section in this DevOps tutorial and check out the DevOps life cycle. This way, we can understand how DevOps works.

DevOps Lifecycle

DevOps focuses on bringing all the development, operations, and IT infrastructure guys, including Developers, Testers, System Admins, and QAs, under one roof. Hence, all these people together are called **DevOps Engineers**.

DevOps Engineers share the end-to-end responsibility of gathering information, setting up the infrastructure, developing, testing, deploying, continuous monitoring, and fetching feedback from end-users. This process of developing, testing, deploying, and monitoring keeps on repeating for better results.


PLAN
CODE
BUILD
TEST
RELEASE
DEPLOY
MONITOR
OPERATE

- **Code:** The first step in the DevOps life cycle is coding, where developers build the code on any platform

- **Build:** Developers build the version of their program in any extension depending upon the language they are using

- **Test:** For DevOps to be successful, the testing process must be automated using any automation tool like Selenium

- **Release:** A process for managing, planning, scheduling, and controlling the build in different environments after testing and before deployment

- **Deploy:** This phase gets all artifacts/code files of the application ready and deploys/executes them on the server

- **Operate:** The application is run after its deployment, where clients use it in real-world scenarios.

- **Monitor:** This phase helps in providing crucial information that basically helps ensure service uptime and optimal performance

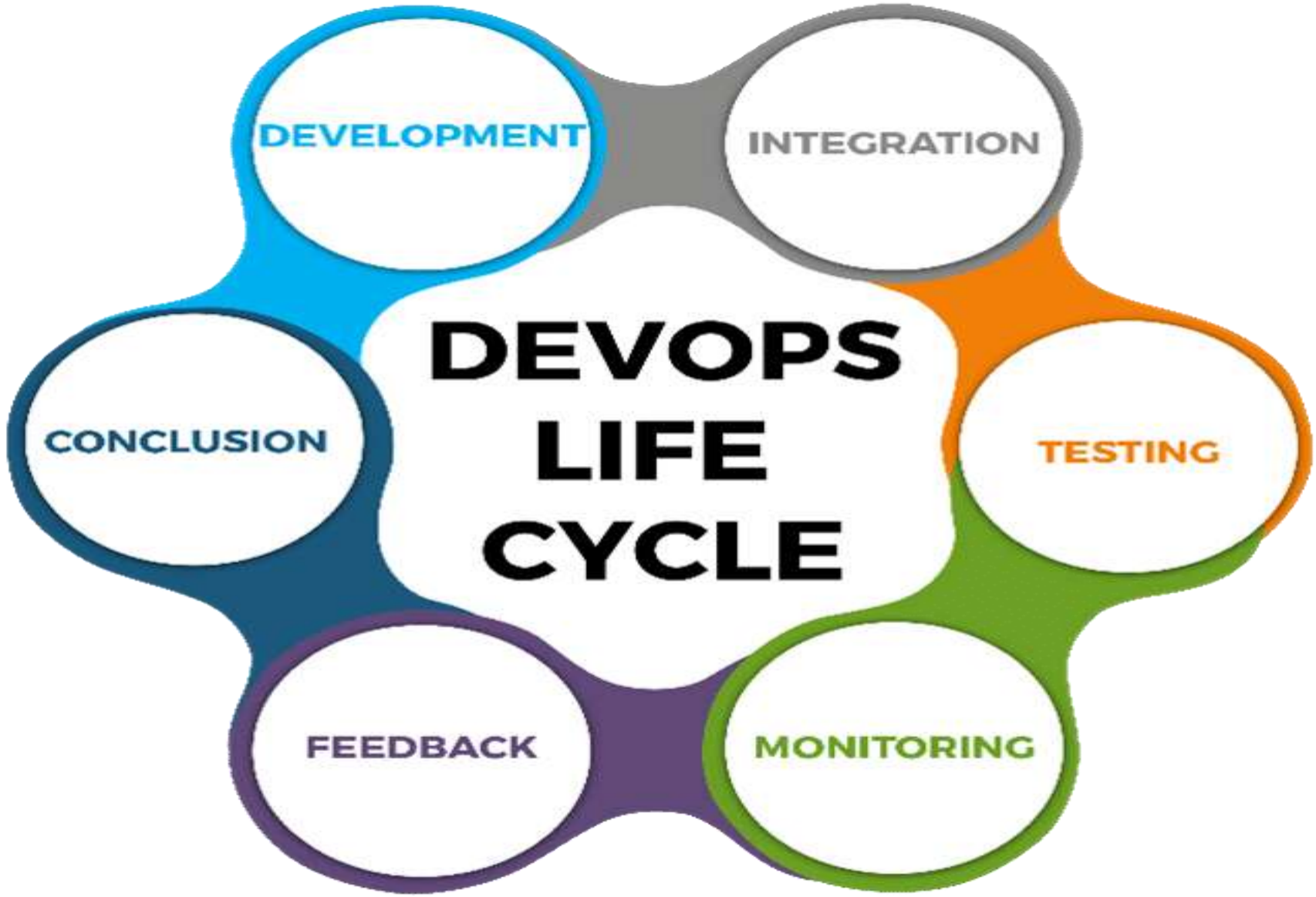
- **Plan:** The planning stage gathers information from the monitoring stage and, as per feedback, implements the changes for better performance

Different Lifecycle Stages

Now, let's discuss the different stages in the DevOps life cycle that contributes to the consistent software development life cycle (SDLC):

- Continuous Development
- Continuous Integration
- Continuous Testing
- Continuous Monitoring
- Virtualization and Containerization

These stages are basically the aspects for achieving the DevOps goal.


DEVELOPMENT
INTEGRATION
CONCLUSION
DEVOPS
LIFE
CYCLE
TESTING
FEEDBACK
MONITORING

Now, let's discuss each of them in detail.

Continuous Development

In the Waterfall model, our software product gets broken into multiple pieces or sub-parts for making the development cycles shorter, but in this stage of DevOps, the software is getting developed continuously.

•**Tools used**: As we code and build in this stage, we can use **GIT** to maintain different versions of the code. To build/package the code into an executable file, we can use a reliable tool, namely, **Maven**.

Continuous Integration

In this stage, if our code is supporting a new functionality, it is integrated with the existing code continuously. As the continuous development keeps on, the existing code needs to be integrated with the latest one '**continuously**,' and the changed code should ensure that there are no errors in the current environment for it to work smoothly.

•**Tools used**: **Jenkins** is the tool that is used for continuous integration. Here, we can pull the latest code from the GIT repository, of which we can produce the build and deploy it on the test or the production server.

Continuous Testing

In **continuous testing** stage, our developed software is getting tested continuously to detect bugs using several automation tools.

•**Tools used:** For the **QA/Testing** purpose, we can use many automated tools, and the tool used widely for automation testing is Selenium as it lets QAs test the codes in parallel to ensure that there is no error, in competencies, or flaws in the software.

Continuous Monitoring

It is a very crucial part of the DevOps life cycle where it provides important information that helps us ensure service uptime and optimal performance. The operations team gets results from reliable monitoring tools to detect and fix the bugs/flaws in the application.

•**Tools used**: Several tools such as **Nagios, Splunk, ELK Stack**, and **Sensu** are used for monitoring the application. They help us monitor our application and servers closely to check their health and whether they are operating actively. Any major issue detected by these tools is forwarded to the development team to fix in the continuous development phase.