Parshvanath Charitable Trust's

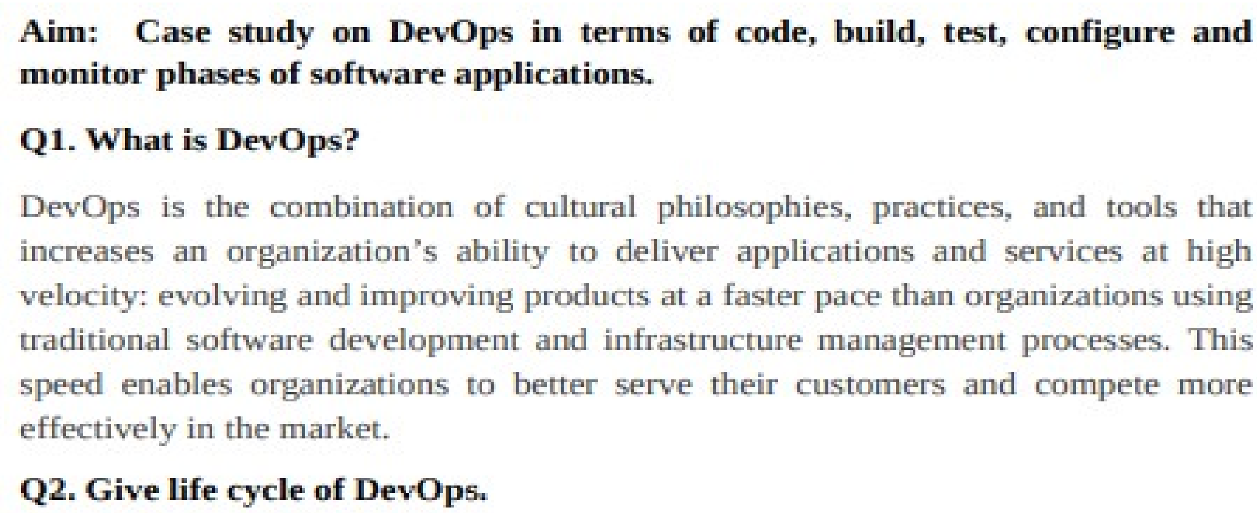
A. P. SHAH INSTITUTE OF TECHNOLOGY

(Approved by AICTE New Delhi & Govt. of Maharashtra, Affiliated to University of Mumbai)
(Religious Jain Minority)

Department of Information Technology

Academic Year: 2020-21	**Name of Student: Om Bheda**
Semester: VIII	**Student ID:19204011**
Class / Branch: IT	**Date of Performance:20/01/22**
Subject: DevOps Lab	**Date of Submission:20/01/22**
Name of Instructor:	

Experiment No. 1

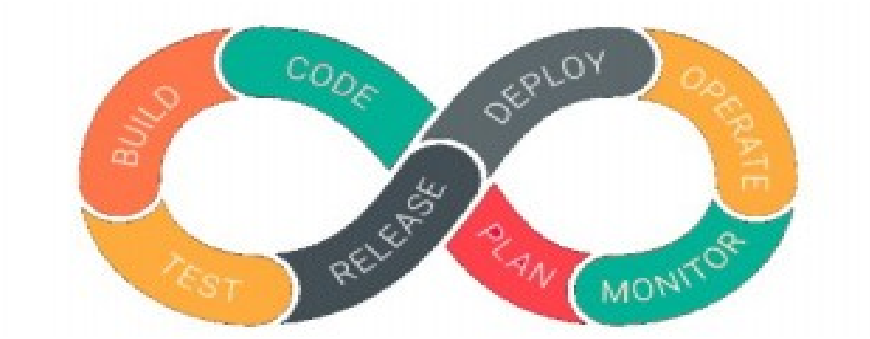


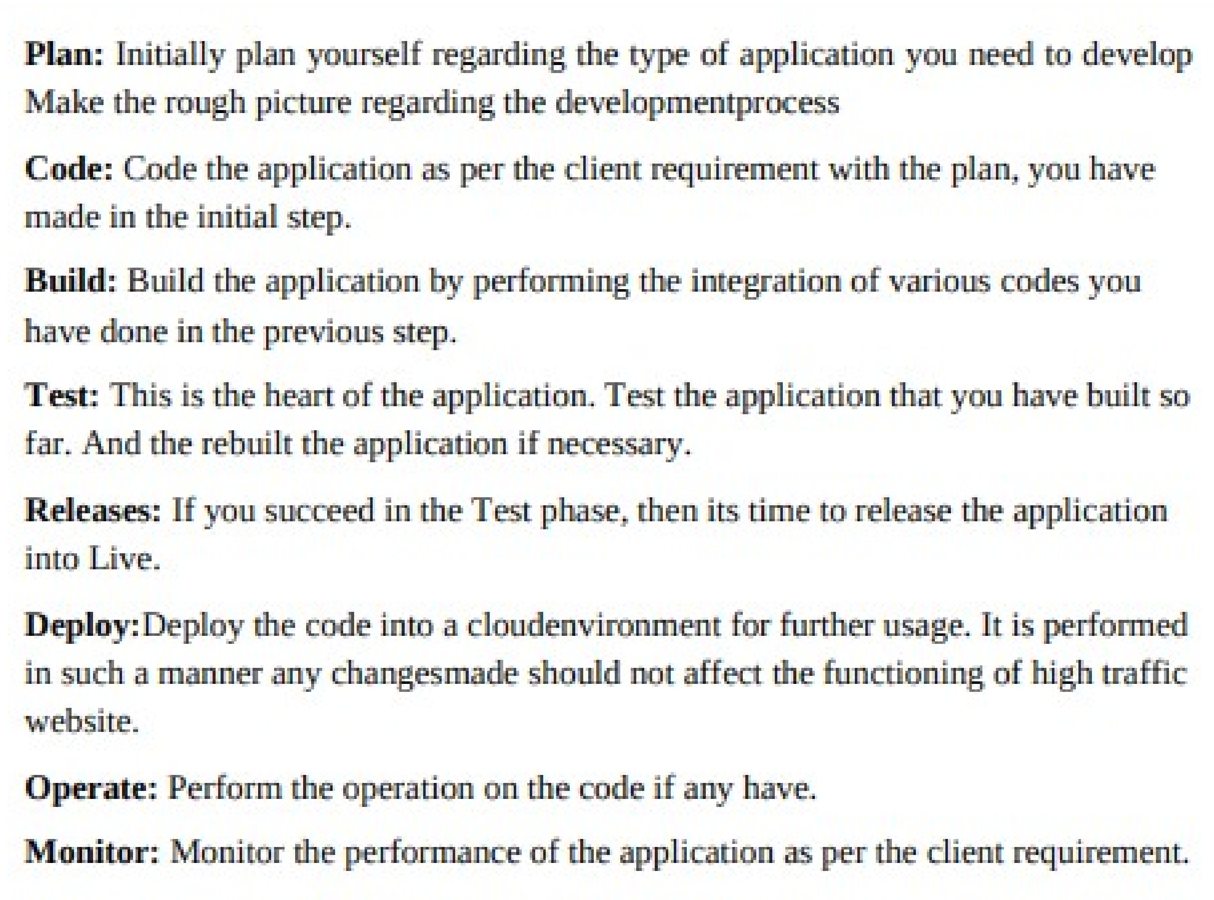
Aim: Case study on DevOps in terms of code, build, test, configure and monitor phases of software applications.

Q1. What is DevOps?

DevOps is the combination of cultural philosophies, practices, and tools that increases an organization's ability to deliver applications and services at high velocity: evolving and improving products at a faster pace than organizations using traditional software development and infrastructure management processes. This speed enables organizations to better serve their customers and compete more effectively in the market.

Q2. Give life cycle of DevOps.





Plan: Initially plan yourself regarding the type of application you need to develop Make the rough picture regarding the developmentprocess

Code: Code the application as per the client requirement with the plan, you have made in the initial step.

Build: Build the application by performing the integration of various codes you have done in the previous step.

Test: This is the heart of the application. Test the application that you have built so far. And the rebuilt the application if necessary.

Releases: If you succeed in the Test phase, then its time to release the application into Live.

Deploy: Deploy the code into a cloudenvironment for further usage. It is performed in such a manner any changesmade should not affect the functioning of high traffic website.

Operate: Perform the operation on the code if any have.

Monitor: Monitor the performance of the application as per the client requirement.

Q3. Explain different phases of DevOps Life cycle.

I) Continuous Development

This is the phase that involves ‘planning‘ and ‘coding‘ of the software. The vision of the project is decided during the planning phase and the developers begin developing the code for the application. There are no DevOps tool that are required for planning, but there are a number of tools for maintaining the code.

II) Continuous Testing

This is the stage where the developed software is continuously tested for bugs. For continuous testing, automation testing tools like Selenium, TestNG, JUnit, etc are used. These tools allow QAs to test multiple code-bases thoroughly in parallel to ensure that there are no flaws in the functionality. In this phase, Docker Containers can be used for simulating the test environment.

III) Continuous Integration

This stage is the heart of the entire DevOps life cycle. It is a software development practice in which the developers require to commit changes to the source code more frequently. This may be on a daily or a weekly basis. Every commit is then built and this allows early detection of problems if they are present. Building code not only involves compilation but it also includes code review, unit testing, integration testing, and packaging.

IV) Continuous Deployment

This is the stage where the code is deployed to the production servers. It is also important to ensure that the code is correctly deployed on all the servers. Before moving on, let us try to understand a few things about Configuration management and Containerization tools. These set of tools here help in achieving Continuous Deployment (CD).

V) Continuous Monitoring

This is a very crucial stage of the DevOps life cycle where you continuously monitor the performance of your application. Here vital information about the use of the software is recorded. This information is processed to recognize the proper functionality of the application. The system errors such as low memory, server not reachable, etc are resolved in this phase.

VI) Continuous Feedback

The application performance is improved consistently by analyzing the final outcome of the product. The continuous feedback is an important phase of the software application where customer feedback is a big asset to improve the working of the current software product and release new versions quickly based on the response.

VII) Continuous operations

All DevOps operations are based on continuity with complete automation of the release process and allow organizations to accelerate the overall time to market on an ongoing basis. It is clear from the discussion that continuity is the critical factor in DevOps removing the abundant steps that often distract the development, take it longer to detect issues, and producing a better version of the product after several months. With DevOps, you can make any software product more efficient and increase the overall count of interested customers in your product. Let us see how businesses are benefitted through DevOps deployment.

Q4. Explain different DevOps tools with its functionalities.

I) Gradle

Gradle

Gradle is an incredibly versatile tool which allows you to write your code in Java, C++, Python, or other languages. Gradle is also supported by popular IDEs such as Netbeans, Eclipse, and IntelliJ IDEA. If that doesn't convince you, it might help to know that Google also chose it as the official build tool for Android Studio.

II) Git

Git

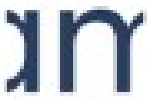
Gitis one of the most popular DevOps tools, widely used across the software industry. It's a distributed SCM (source code management) tool, loved by remote teams and open source contributors. Git allows you to track the progress of your development work. You can save different versions of your source code and return to a previous version when necessary. It's also great for experimenting, as you can create separate branches and merge new features only when they're ready to go.

III) Jenkins

Jenkins

Jenkins is the go-to DevOps automation tool for many software development teams. It's an open source CI/CD server that allows you to automate the different stages of your delivery pipeline. The main reason for Jenkins' popularity is its huge plugin ecosystem. Currently, it offers more than 1000 plugins, so it integrates with almost all DevOps tools, from Docker to Puppet. With Jenkins, you can set up and customize your CI/CD pipeline according to your own needs

IV) Bamboo



Bamboo

Bamboo is Atlassian's CI/CD server solution that has many similar features to Jenkins. Both are popular DevOps tools that allow you to automate your delivery pipeline, from builds to deployment. However, while Jenkins is open source, Bamboo comes with a price tag. So, here's the eternal question: is it worth choosing proprietary software if there's a free alternative? It depends on your budget and goals. Bamboo has many pre-built functionalities that you have to set up manually in Jenkins. This is also the reason why Bamboo has fewer plugins (around 100 compared to Jenkins' 1000+). In fact, you don't need that many plugins with Bamboo, as it does many things out-of-the-box.

V) Docker

Docker

Docker integrates with Jenkins and Bamboo, too. If you use it together with one of these automation servers, you can further improve your delivery workflow. Besides, Docker is also great for cloud computing. In recent years, all major cloud providers such as AWS and Google Cloud added support for Docker. So, if you are planning a cloud migration, Docker can ease the process for you.

VI) Kubernetes

Kubernet...

Kubernet is a container orchestration platform that takes containerization to the next level. It works well with Docker or any of its alternatives. Kubernetes is still very new; its first r release came out in 2015. It was founded by a couple of Google engineers who wanted to find a solution to manage containers at scale. With Kubernetes, you can group your containers into logical units.

VII) Ansible

Ansible is a configuration management tool, similar to Puppet and Chef. You can use it to configure your infrastructure and automate deployment. Its main selling points compared to other similar DevOps tools are simplicity and ease of use. Ansible follows the same Infrastructure As Code (IAC) approach as Puppet. However, it uses the super simple YAML syntax. With

IV) Bamboo

Bamboo

Bamboo is Atlassian's CI/CD server solution that has many similar features to Jenkins. Both are popular DevOps tools that allow you to automate your delivery pipeline, from builds to deployment. However, while Jenkins is open source, Bamboo comes with a price tag. So, here's the eternal question: is it worth choosing proprietary software if there's a free alternative? It depends on your budget and goals. Bamboo has many pre-built functionalities that you have to set up manually in Jenkins. This is also the reason why Bamboo has fewer plugins (around 100 compared to Jenkins' 1000+). In fact, you don't need that many plugins with Bamboo, as it does many things out-of-the-box.

V) Docker

Docker

Docker integrates with Jenkins and Bamboo, too. If you use it together with one of these automation servers, you can further improve your delivery workflow. Besides, Docker is also great for cloud computing. In recent years, all major cloud providers such as AWS and Google Cloud added support for Docker. So, if you are planning a cloud migration, Docker can ease the process for you.

VI) Kubernetes

Kubernet...

Kubernet is a container orchestration platform that takes containerization to the next level. It works well with Docker or any of its alternatives. Kubernetes is still very new; its first r release came out in 2015. It was founded by a couple of Google engineers who wanted to find a solution to manage containers at scale. With Kubernetes, you can group your containers into logical units.

VII) Ansible

Ansible is a configuration management tool, similar to Puppet and Chef. You can use it to configure your infrastructure and automate deployment. Its main selling points compared to other similar DevOps tools are simplicity and ease of use. Ansible follows the same Infrastructure As Code (IAC) approach as Puppet. However, it uses the super simple YAML syntax. With