Birla Institute of Technology and Science, Pilani

INTRODUCTION TO DEVOPS

Assignment – Critique Report

Group No - 3

Topic: DevOps Project

No.	Team Members	Roll No
1	Nitin Kumar	2020HS70003
2		
3		
4		
5		

Table of Contents

Table of Contents ***2***

Critique Report ***3***

DevOps Practises: ***3***

Challenges and Gaps ***3***

Enhancement to current practise ***3***

Critique Report

DevOps Practises:

DevOps Practises followed in the team:

1. GitHub Workflows: We opted for Feature branch workflow, where every feature was developed in a new branch and upon basic sanity checks it was allowed to merge with master.

2. Continuous Integration: Since the very beginning we emphasised on maintaining Code Quality. For that we used GitHub Actions to automate our build for every single Branch, Pull Requests and Commits. Along with this we also added an additional phase in the build to integrate with SonarQube cloud offering called SonarCloud. Upon basic checks like whether build is passing or not, is there any vulnerable code, or bugs in a PR or not. After these a PR was merged.

3. DevOps Mindset: In case on any wrong merge causing the build or pipeline to fail, our highest priority was to fix the build and then implement any new feature.

4. Automated Testing: While integrating the frontend with backend we identified a few bugs, later upon fixing the reported bug, due to Unit Tests in place we were able to find a new bug caused by the fix. Though the build was green, but this mutation was cached due to having automated tests.

5. Pipeline As A Code: Having this we were easily able to switch out local Jenkins instance to an Amazon EC2 instance. However due to poor compute power we had to switch back to the local instance. This wouldn't have been possible without PAAC, as for us it was matter of minutes to setup an EC2 instance, install Jenkins and create job using Jenkinsfile in the SCM (GitHub)

Challenges and Gaps

1. Microservices Architecture: Having in total 5 backend services and 2 frontend service, it was quite challenging to do the initial setup and also the learning curve was a little steep.

   During dockerization all the service it was very difficult to setup communication between the services in a container environment and due to various type of communication available. Upon read the purpose we figured out the best suited for service to service communication, service to API Gateway communication and Service to Eureka Discovery client communication.

2. Deploying: Deploying with microservices is not easy and we choose the route using docker-compose, instead using K8s, deploying to a K8s cluster would have been an ideal choice, but due to gaps in the knowledge and time we were not able to figure it out.

Enhancement to current practise

1. Externalizing the configuration using Cloud-Config: Currently the configuration is not externalized, it is passed as a parameter, if not present then read from environment variable. Externalizing this would make our job easily when running in local environment vs docker-containers.

2. Fully using the potential of a Multi Branch Pipeline: Upon introspection it was notice that we made change to Jenkinsfile and merged the code to master then trigger the build, instead we could have simply used the branch used for PR and run the job for that branch.

3. Jenkins As A Service: we used a local instance of Jenkins due to lack of enough free compute power offered by various Cloud IAAS providers.

4. Integrating with various other tools for other purposes.