# Case Studies and Final Exam

—----------------------------------------------------------------------

In these  case studies, you will read and analyze DevOps-based scenarios, apply what you learned in the Introduction to DevOps course, and answer graded quiz questions based on these scenarios.

### **Learning Objectives**

* Apply characteristics of DevOps culture to scenario-based real-world software development issues.
* Apply the concept of organization around business domains and Git repository guidelines to scenario-based real-world software development issues.
* Analyze social coding principles in a scenario-based real-world software development issues.

—----------------------------------------------------------------------------------------------------------------------------

Miguel has just joined the new DevOps team at the Acme company. He has been instructed to work closely with the Dev team and the Ops team as a liaison. In his normal day, he maintains the Continuous Integration/Continuous Delivery (CI/CD) pipelines and makes sure that the developers have what they need from Ops and makes sure that Ops has what they need from the developers. When the Dev team has a request, Miguel uses the ticket queue to make that request to Ops on Dev’s behalf. One morning, the Dev team needed virtual machines (VMs) provisioned in a hurry to deploy their application by the deadline.

Miguel thought he had a good relationship with Charles, the SysAdmin, and asked him if he could get the VMs done that day. Charles curtly reminded him that they had a well-established ticket queue system, and that he would have to open a ticket like everyone else. Miguel opened a ticket and let the developers know that he made the request for the VMs. Several days passed. Nancy, on the Dev team, asked Miguel where the VMs were. She reminded him about the deadline and said, “We’ll miss it unless you get us those VMs.”

Two more days passed. In a meeting with the Dev team, Jim, one of the team members, said, “Well, I do kind of have access to something.” He explained that he had a personal Cloud account where he could have the VMs provisioned within the hour. Miguel asked whether this was in concert with corporate policy. Jim replied, “We have to do what it takes to meet customer commitments.” They deployed the application on Jim’s cloud account.

***After reading this scenario, consider what you have learned in the course and apply that knowledge to the quiz questions that follow.***

***—-----------------------------------------------------------------------------------------------------------------------------***

*Well, in this scenario I noticed a lot of mistakes.*

*First: The company has a wrong understanding of DevOps, the company already has an operations department and a development department, they think that when they create a new DevOps department, it will improve the work environment, but the exact opposite happened,*

*Because first, an important principle of DevOps is not to work in silos, so I think first that the company should change its way of thinking and adopt the DevOps culture and apply DevOps principles to the company and the different teams.*

*Second: Both development teams and operations teams should work together, apply the principles of Agile and the different principles of DevOps, and share the same goal together and have a culture of openness, transparency and trust with each other. This will enhance effective communication between development teams and operations teams, which leads to effective communication and early detection of errors, because detecting errors in the late stages such as the testing stage is very costly for the program or application production plan.*

*Also, all teams such as operations and development should adopt the DevOps culture and share code together. This will help in the process of discovering errors early, which will also help in creating multiple solutions for a better solution in the end.*

*This is my point of view in this scenario*

***Ahmed Zakaria***

***—----------------------------------------------------------------------------------------------------------------------------***

# *Scenario 2 Organizing DevOps*

*Estimated time needed: 5 minutes*

*Roopa is an application developer at a company that is building their own e-commerce website. She just picked up a story from the Kanban board to add a new feature that requires input from the user. Since she is a back-end developer, she must coordinate with the front-end team to add new data fields to the user interface. She opens an issue on the front-end team’s GitHub to add the required fields. Her feature requires a schema change (that is, a change to the table structure in the database) to hold the new data. So, she opens a ticket with the database team to make the change.*

*She works on the application logic for her feature and checks the code into the development branch in GitHub. Unfortunately, she can’t test her code until the front-end team updates the user interface (UI) and the database team updates the schema. Because she is blocked, she begins working on another feature story.*

*Two weeks later, the front-end team and the database team make their changes. Roopa stops working on the feature she is currently working on to go back and merge those changes with her previous code. Her attempt to merge encounters several merge conflicts. It takes several hours to resolve the merge conflicts because the code has drifted over time. She commits her updated code to the development branch. At the end of the month, the development team merges the development branch back into the master branch to prepare for release. Unfortunately, some quick bug fixes that were applied to the master branch now conflict with Roopa’s code in the development branch. She has to spend an entire day remediating those conflicts.*

*The development team gets everything working and they are ready to deploy to production. They manually deploy the new release into production only to have it fail. The team had to roll back the release so they could determine the cause of the failure. They meticulously went through every feature change to find the cause of the failure. It took three days to test each new feature in isolation. They determined that the schema update for Roopa’s new feature was never applied. Once the schema was updated, everything started working. Roopa overhears a manager complain about how long this took. One developer shrugs and says, “It’s pretty much always like this.”*

***After reading this scenario, consider what you have learned in the course and apply that knowledge to the quiz questions that follow.***

***—----------------------------------------------------------------------***

***My answer***

*Well, my point of view in this scenario is that it is very slow in different departments, such as the operations department, the front-end graphic department, and the database department.*

*I see a big problem in this scenario, which is that each of these departments works individually in silos, and each department waits for the other department to update the data, and this is completely wrong, this approach will definitely cause the production process to slow down, which will delay the delivery process to customers, which reflects on the company's reputation, this approach is very destructive.*

*The solution is simple, all teams should adopt the DevOps culture, all these teams should be integrated and work together, and share problems together, communication between them will create a kind of good and effective communication, and this will help in the process of discovering errors very early, and will also open the door to different opinions, which will help in the best solution for each problem, and certainly after effective communication, the process of producing the program will be completed in a short time, as well as the process of sharing code, and testing, these teams should feel that they are one entity and that each one of them is responsible for the final product, and this will enhance the feeling of each individual in the teams, and will give them the right to the process of continuous testing, and solving problems continuously, which will help in the process of continuous development / and the process of continuous delivery to customers.*

*This is the solution from my point of view for this scenario*

*Ahmed Zakaria.*

***—----------------------------------------------------------------------------------------------------------------------------***

# ***Scenario 3 Social Coding***

*Estimated time needed: 5 minutes*

*Beta company has several teams that maintain their e-commerce website. They have adopted the microservices architecture and organized their teams along business domains so that they have an account team, a product team, an orders team, and a warehouse team.*

*The account team has decided to give customers a more personalized experience. They want to be able to recommend products a customer might like based on what other customers are buying. They notice that the product team has a similar function for recommending products and accessories that go with what the customer is viewing online.*

*Jeff, the account team lead, decided to ask Susan from the product team if they could add this new capability to their existing recommendation function. Susan said the product team was very busy and, “It will be about a month before we can get to it.” Jeff asked whether he could make the changes. Susan replied, “Our code repository is private, and we don’t like to give access to people outside our team.”*

*Jeff didn’t want to wait and instead, had the account team write 100% of the code from scratch. He didn’t think it would be that difficult and that they would be able to do it faster than waiting for the product team to do it. Unfortunately, it was harder than Jeff expected.*

*Jeff knew Kiet, one of the most experienced product team members, from a previous project and asked if he could help them with the new feature. Kiet spent two weeks reviewing the account team’s code with them and making suggestions that got the account team back on track.*

*The account team eventually completed the feature. They delivered two weeks later than anticipated, but because they created a completely new feature, each of the five account team members received an Innovative Contribution Award that included a $500 bonus.*

*Kiet, who never received any credit for his contribution thought to himself, “That’s the last time I help the account team with anything.”*

***After reading this scenario, consider what you have learned in the course and apply that knowledge to the quiz questions that follow.***

In this scenario, unfortunately, a big problem is the lack of sharing of code between different teams.

Although it is a good thing to have different teams, each specialized in solving a specific part of the product, there is another very important thing that must be applied, which is / working together.

All these teams should not work in separate silos, and should work together together, and this will enhance collective thinking and invent creative solutions and more features, and this will return to all teams with increase and bonus, and also a very important thing is that they should share the code between all these teams.

A very common fallacy is that each team thinks that this code is ours and no one can see it. Unfortunately, this is a wrong thinking that should not be applied. On the contrary, the code should be shared between these different teams, to complete tasks faster. This will also enhance the time and will help the company produce the program in a much shorter time than expected. Also, when they share the code, there will be more than one eye that sees the code, and this will enhance the process of discovering errors in the early stages, and will also help in finding different and more effective solutions.

This is my point of view in this scenario.

**Ahmed Zakaria**