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.

Author(s)

- John Rofrano
- Mary Deville