Staging vs. Production

Development Environments

Every development team prior to releasing their new features or changes needs to verify that the code they do release is not going to cause any issues or bugs. In order to achieve this, they normally set up multiple environments for different ways to test and verify. A common practice is for teams to have a developer environment, a UAT or QA environment, and a staging environment. The main purpose of this flow is to find any potential issues that may arise due to changes or new features being added to the codebase. The more ways to test the changes the less likely bugs will be introduced.

Staging

The staging environment should mimic your production environment. The reason for this is because you want to test the code in an environment that matches what you have in production. This allows teams to spot or find any potential issues prior to them getting to production. The closer the staging environment is to your production, the more accurate your testing is going to be. Staging environments can also be used for testing and verifying new features and allow other teams including QA or stakeholders to see and use those features as a pre-trial. Staging should also cover all areas of the architecture of the application including the database and any other services that may be required. Areas that benefit from staging environments include:

New Features

Developers submitting new features along with feature flags for turning them on and off should always do a testing round in a staging environment. They allow teams to verify that the feature works, it can be turned on and off via configuration flags and also that it does not break or interfere with existing functionality.

Testing

As the staging environment mimics your production environment, it's also a great place to run tests. QA teams will normally use it to verify new features, configuration changes or software updates/patching. The types of testing covered will be Unit testing, Integration testing and performance testing. All except performance testing can also be carried out in production. Performance can also be completed in production but only at specific times - usually out of hours as it will have a drastic effect on the user experience.

Sometimes it is not always feasible to have an exact replication either due to costs or time. Certain areas can be cut back - for example, if your service is load balanced on 10 virtual

machines in production, you could still have 4 virtual machines in staging. The underlying architecture is the same but the overall performance may be different.

Migrations

Staging is a perfect place to test and verify data migrations. Snapshots can be taken from production and used to test your migration scripts to confirm your changes will not break anything. If in the case it does cause an issue, you simply rollback and try again. Doing something like a migration in production is extremely risky and error-prone.

Configuration Changes

Configuration can also cause headaches for teams, especially in a large cloud-based architecture. Having a staging environment will allow you to spot any potential issues or bottlenecks.

Production

Production is live. It's out there for people to see and/or interact with. Any issues or problems you may have had should have been caught and fixed in the staging environment. The staging area gives the team a safety net to catch these possible issues. Any code that is deployed to production should have been tested and verified before the deployment itself.

Downtime

Downtime for any service especially customer facing will most likely be revenue impacting. If customers can not access or use your website or app to its full capabilities, it will most likely have a cost involved. Take for example an e-commerce company that allows users to buy goods and services online. If they release a new feature to their shopping cart which actually breaks the payment process, this will have an impact on customers not being able to buy goods online.

Vulnerabilities

Cyber-security should also play a big role in what gets released in production. Any updates to software such as patching or moving to the latest version should be checked and verified. This is also the same rule for not upgrading software when critical updates are released.

Reputation

Downtime or issues in production is damaging for a company as it does not instill confidence in end users. If something is down or broken it can cause the company to lose potential customers.