Step 1: Review the customer case study

Outcome

Analyze your customer needs.

Timeframe

15 minutes

Fabrikam Medical

Fabrikam Medical Conferences provide conference website services tailored to the medical community. Over the course of ten years, they have built conference sites for a small conference organizer. Through word of mouth, Fabrikam Medical Conferences has become a well-known industry brand handling over 100 conferences per year and growing.

The VP of Engineering at Fabrikam, Susan Withers, has a team of 12 developers who handle all aspects of development, testing, deployment, and operational management of their customer sites. Due to customer demands, they have issues with the efficiency and reliability the conference websites. This mainly caused by an inefficient development and operations workflow.

In the current situation, the conference sites are hosted on-premises with the following topology and platform implementation:

The conference web sites are built with the MEAN stack (Mongo, Express, Angular, Node.js)

Web sties and APIs are hosted on Linux machines.

MongoDB is also running on a separate cluster of Linux machines.

Azure

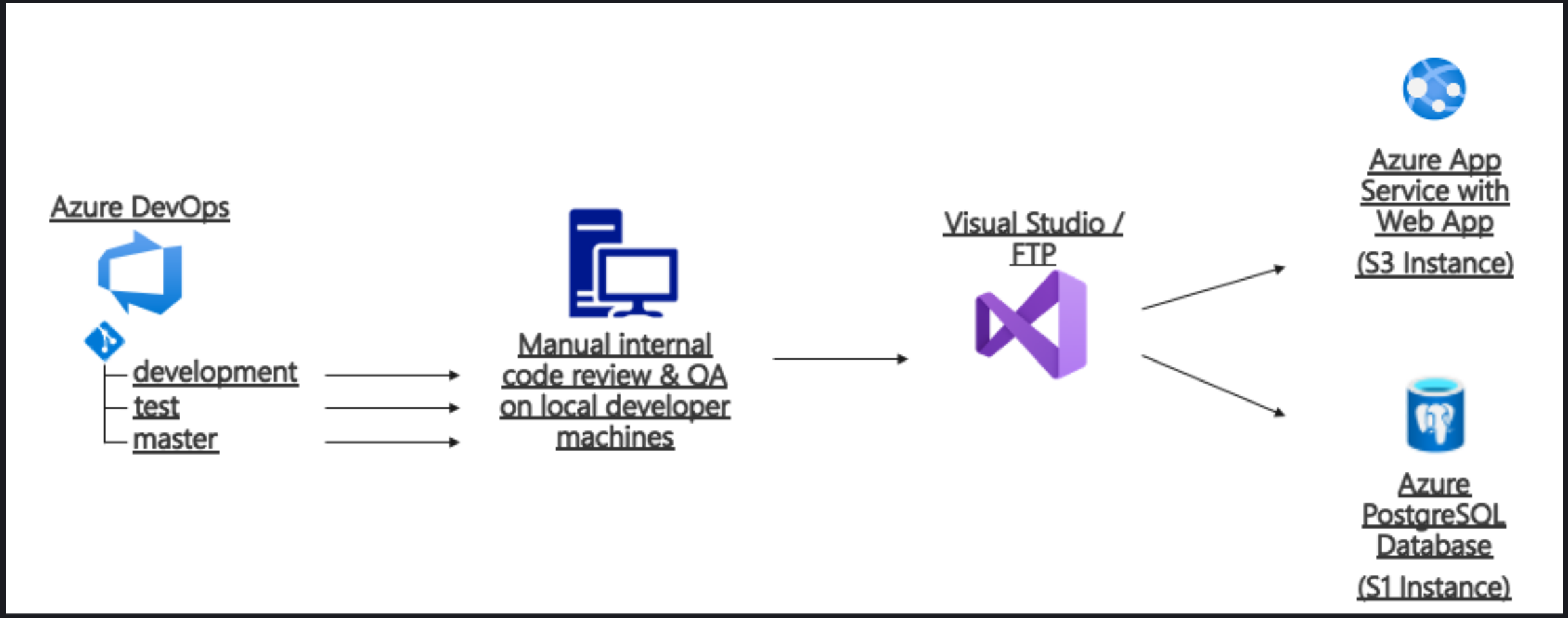
Fabrikam Medical

Needs

- Reduce the overhead in time, complexity, and cost for deploying new conference tenants.
- Improve the reliability of conference tenant updates.
- Choose a suitable platform for their conference solution.
- Migrate data from MongoDB on-premises to CosmosDB with the least change possible to the application code.
- Make use of GitHub and/or Azure DevOps for development lifecycle.
- Use Git repositories for source control and integration into a CI/CD workflow.
- Embed code review and security scanning in to the development workflow.
- Do not incur a direct vendor lock-in.

Fabrikam Medical Conferences

Current Process



Customer objections

- We do not want to be locked into a specific source control repository. We are evaluating GitHub and Azure DevOps and need to be able to change between them without frustrating rework.

- We do not want the developers to be able to make changes to the Azure resources even though they will have access to make source code changes.

- If developers can deploy directly to the cloud, will that expose us to the same quality problems we had before when untested code was promoted to production?

Customer objections

- How much of an impact will these process changes have on our development cadence? Will learning this place a new burden on the developers?

- Our developers are already having challenges learning how to use Git; will adding a continuous deployment system on top of that slow them down and confuse them even more?

Common scenarios