PROMISES AUTO-DELIVERED

CONTINUOUS INTEGRATION/DELIVERY/DEPLOYMENT

ABOUT ME

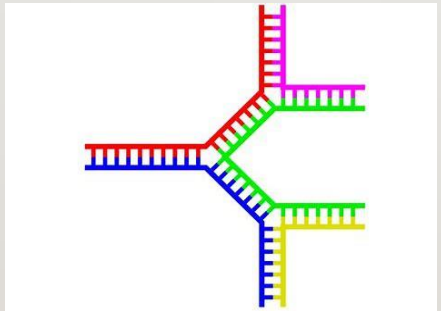
- **Shobana Krishnamoorthy** is a Senior Developer in the Office division at Microsoft Corporation, USA. She has over 12+ years' experience in the technology industry with a background education in Computer Science. As a Developer at Microsoft, Shobana and team are responsible for automating deployments, SharePoint feature management, SharePoint site reliability and security of cloud network end to end starting from requirement to release. She holds five patents in Cloud management and patching. She is a passionate stakeholder of the Women's mentoring circle at Microsoft.
- She was an open source mentor, Anita bee mentor for "Software engineering" track, SOL speaker @GHC 2017, Technical workshop speaker for "Build a plane while flying on it" @ GHC 2018. She has also presented various technical topics on Devops, CI/CD, Dark deployments in Microsoft internal conferences and external like WIT, SWE etc.
- Apart from work, she is an active volunteer @ "Sophia's way", a non-profit organization for assisting women from homeless to Independence, Mentor @ "Tech bridge for girls" – teaching computer science for young girls in middle and high school.

AGENDA

- Continuous Integration/Delivery/Deployment Overview
- Continuous Integration/Delivery/Deployment Techniques
- CI/CD Best Practices
- CI/CD Walkthrough using Azure
- Continuous Integration/Delivery/Deployment (CD) Tools & Comparison
- Questions

TRADITIONAL DELIVERY MODELS != AGILE

Longer development and release cycles



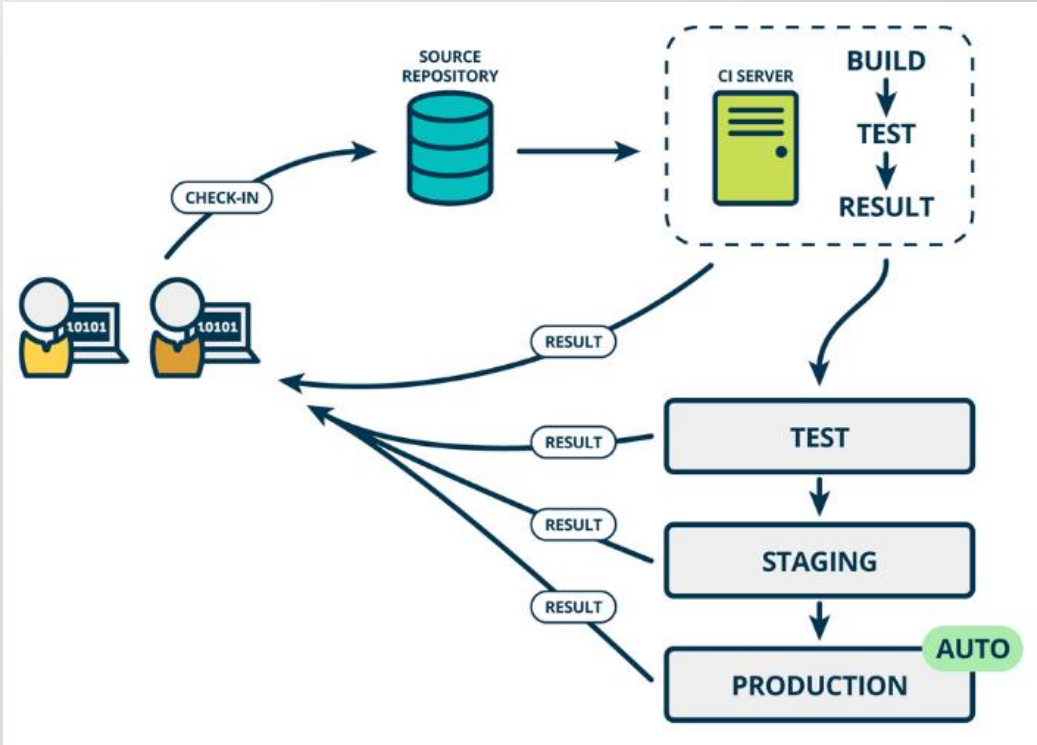
Multiple code branches

Deployment that is tightly coupled with releases

Agile is not a destination, it’s the direction

CONTINUOUS INTEGRATION/DELIVERY/DEPLOYMENT OVERVIEW

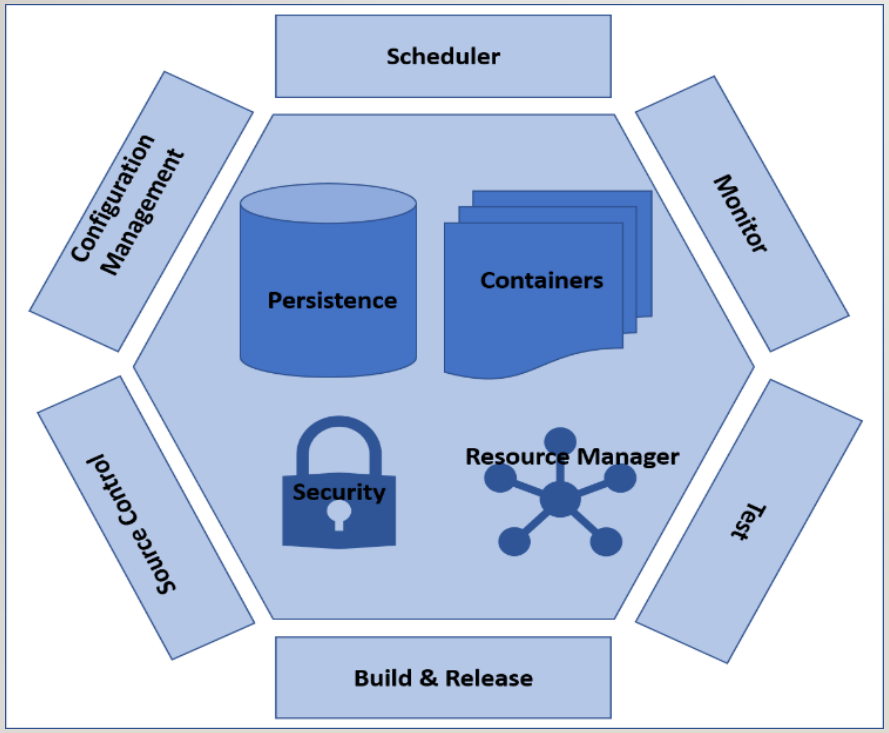
- Continuous Integration is the process of automating the build and testing of code every time when changes are committed to version control.
- Continuous Delivery is the process to build, test, configure and ability to deploy from a build to a production environment. The final decision to deploy to a live production is triggered manually.
- Continuous Deployment is the next step of continuous delivery. Every change that passes the automated tests is deployed to production automatically.



CI/CD BENEFITS

- Automated software release process.
- Improved developer productivity and agility.
- Deploy your code to production at lightning speed with high-performance pipelines and quality. Shorten your cycle time.
- Setup test plans, track and report manual tests, run automated test suites, and cloud-based load tests. Automate tests on thousands of real devices and hundreds of configs in the cloud.
- Continuous and optimized learning - monitor the health of your app, get real-time crash reports and advanced analytics to quickly diagnose and fix problems in beta or production apps.
- Customer satisfaction.

CI/CD ARCHITECTURE



- Persistence
- Containers
- Security
- Resource Manager
- Source Control
- Scheduler
- Configuration management
- Monitor
- Test Pipeline
- Build & Release Pipeline

CI/CD TECHNIQUES

- **Blue-Green Deployments**

Blue-green deployments is a strategy for testing code in a production-like environment and for deploying code with minimal downtime. Two sets of production-capable environments are maintained, and code is deployed to the inactive set where testing can take place. When ready to release, production traffic is routed to the servers with the new code, instantly making the changes available.

- **Branch by Abstraction**

Branch by abstraction is a method of performing major refactoring operations in an active project without long-lived development branches. An abstraction layer is built and deployed between consumers and the existing implementation so that the new implementation can be built out behind the abstraction in parallel.

CI/CD TECHNIQUES

- **Canary Releases**

Canary releases are a strategy for releasing changes to a limited subset of users. The idea is to make sure everything works correctly with production workloads while minimizing the impact if there are problems.

- **Dark launch**

Dark launching is the practice of deploying code to production that receives production traffic but does not impact the user experience. New changes are deployed alongside existing implementations and the same traffic is often routed to both places for testing. The old implementation is still hooked up to the user's interface, but behind the scenes, the new code can be evaluated for correctness using real user requests in the production environment.

CI/CD TECHNIQUES

- **Deployment Pipeline**

A deployment pipeline is a set of components that moves software through increasingly rigorous testing and deployment scenarios to evaluate its readiness for release. The pipeline typically ends by automatically deploying to production or providing the option to do so manually.

- **Feature Flags or Feature Toggles**

Feature flags are a technique of deploying new features behind conditional logic that determines whether or not to run based on the value of an environmental variable. New code can be deployed to production without being activated by setting the flag appropriately. To release the software, the value of the environmental variable is changed, causing the new code path to be activated.

CI/CD TECHNIQUES

- **Promoting**

In the context of continuous processes, promoting means moving a software build through to the next stage of testing.

- **Soak Testing**

Soak testing involves testing software under significant production or production-like load for an extended period of time.

CI/CD BEST PRACTICES

- Keep Your Pipelines Fast. Small and Iterative changes.
- Isolate and Secure Your CI/CD Environment
- Trunk-Based Development
- Make the CI/CD Pipeline the Only Way to Deploy to Production
- Maintain Parity with Production Wherever Possible
- Minimize Branching in Your Version Control System
- Keep the Building and Testing Phases Fast
- Decouple Deployment and Release

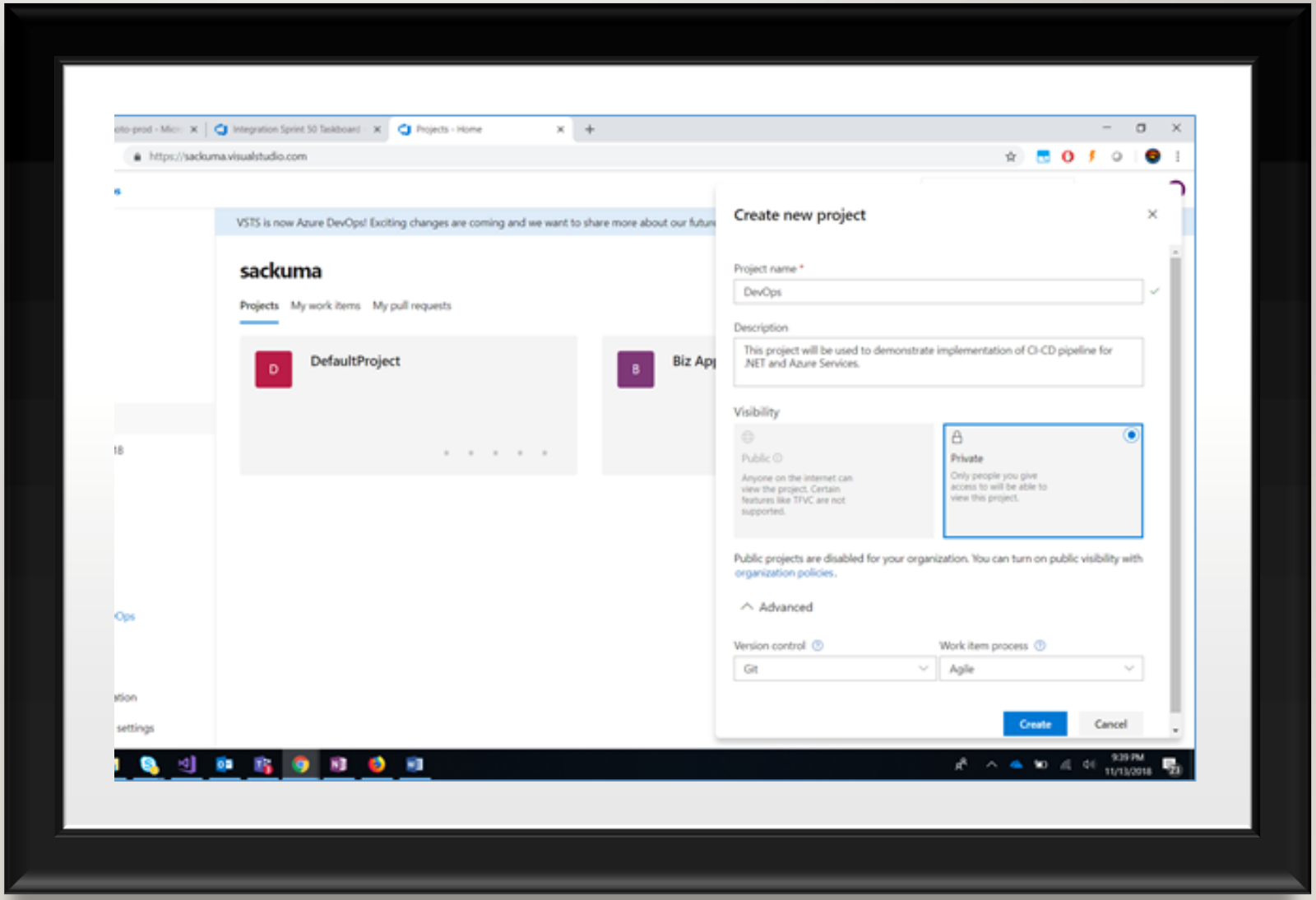
CI/CD WALK THROUGH USING AZURE AND GIT

Steps to set up end-to-end continuous integration and continuous delivery pipeline

1. Set up a Project
2. Define a Branching Strategy / Policies
3. Set up your Repo
4. Set up Builds
5. Set Up Quality Verification Build
6. Set up Release

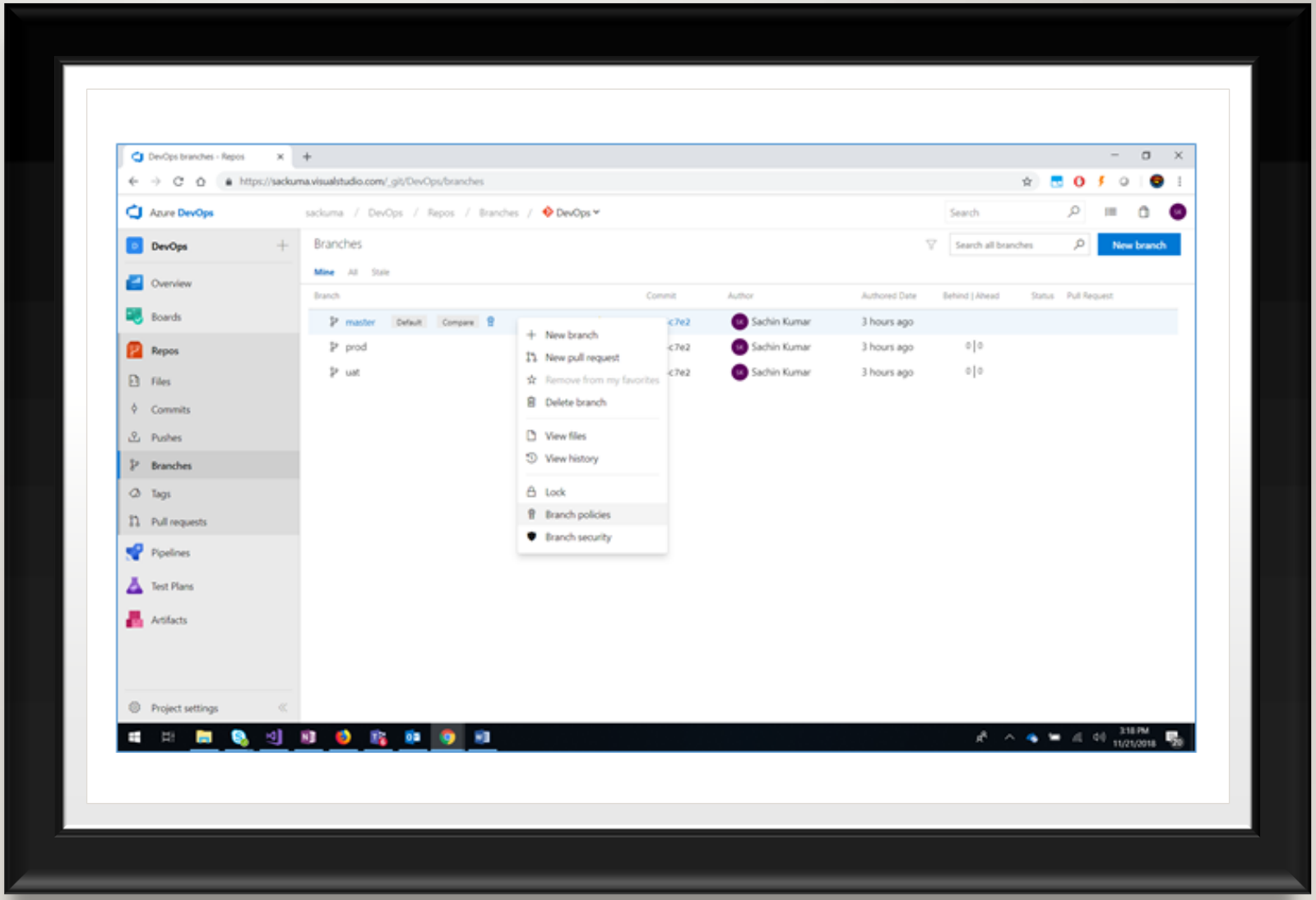
CI/CD WALK THROUGH USING AZURE AND GIT

SET UP A PROJECT



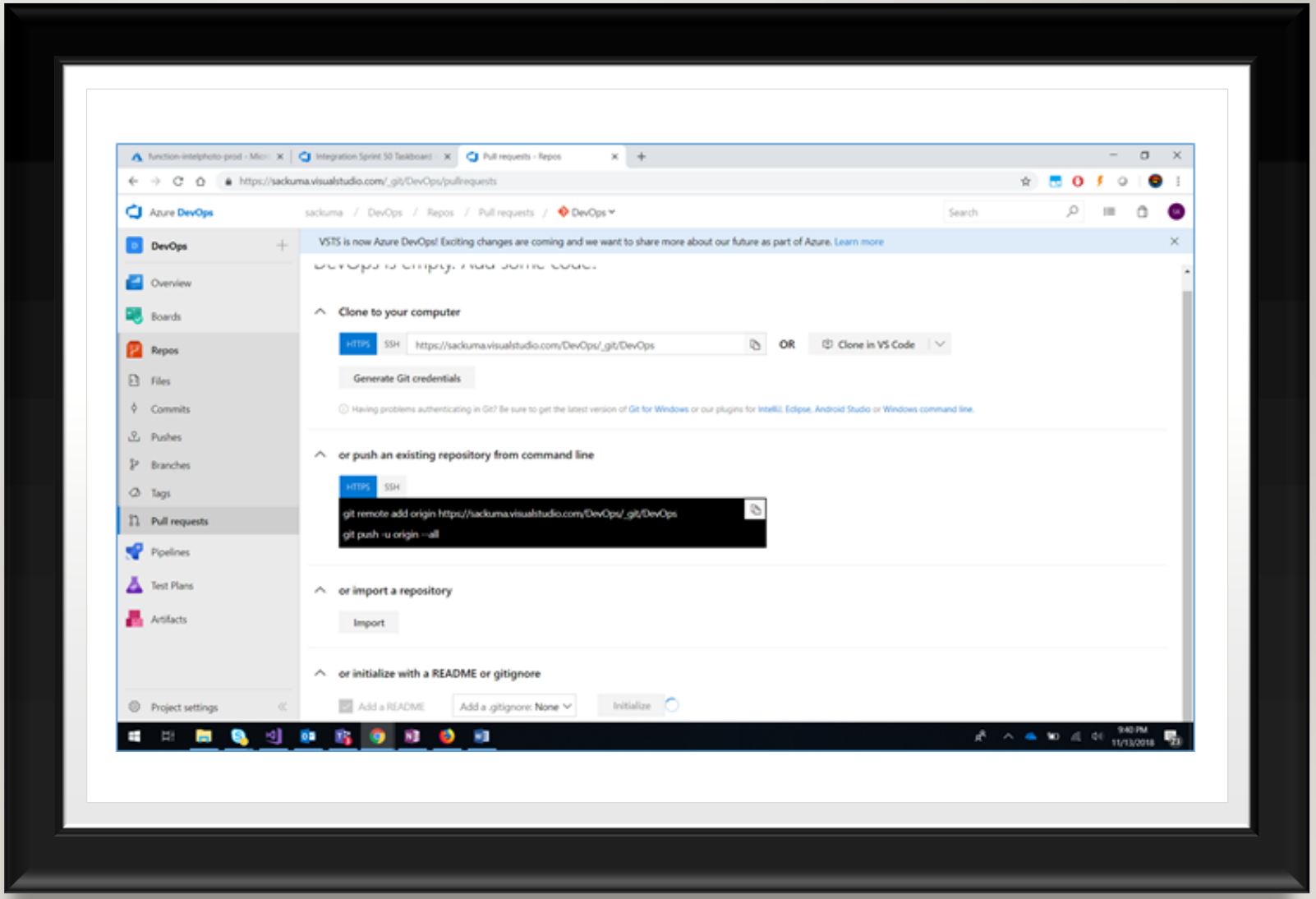
CI/CD WALK THROUGH USING AZURE AND GIT

SET UP BRANCH/POLICIES



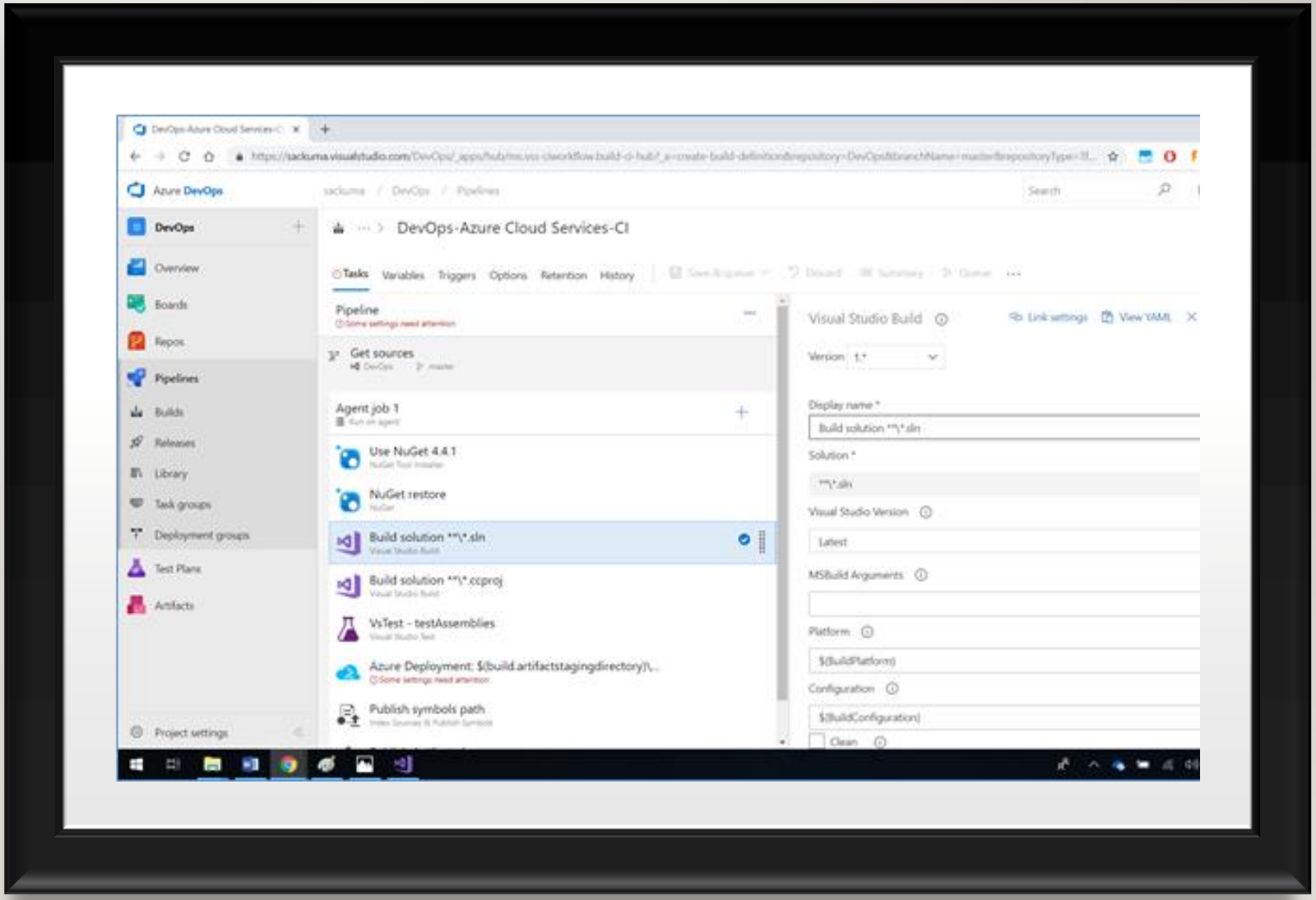
CI/CD WALK THROUGH USING AZURE AND GIT

SET UP A REPO



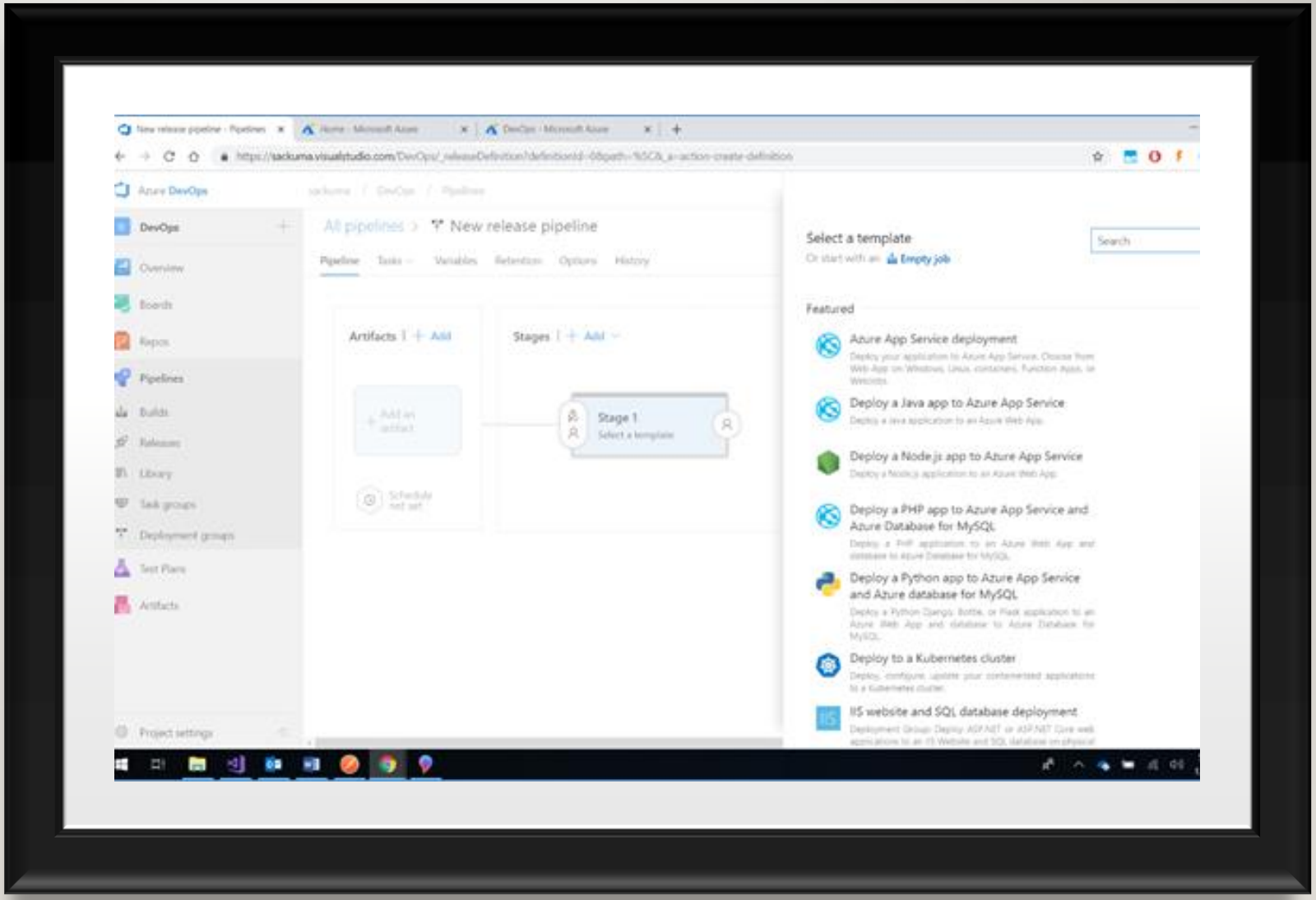
CI/CD WALK THROUGH USING AZURE AND GIT

SET UP A BUILD



CI/CD WALK THROUGH USING AZURE AND GIT

SET UP A RELEASE



CI/CD TOOLS COMPARISON & OPEN SOURCE

- **AWS:** Subscription based, customizable workflows
- **Azure Devops:** Subscription based, git support, customizable workflows
- **Jenkins:** Open source, most popular, web interface & custom plugins, easy installation & deployable across network of machines.
- **Travis**: Open source project on GitHub, bilingual support, works on slack and other notifications, extended API for custom management.

QUESTIONS?