**OpenHack – DevOps 3.0**

**Overview**

**This OpenHack enables attendees to**use DevOps practices to achieve secure zero downtime deployment for microservice based applications running in Azure App Service.

**The OpenHack simulates a real-world scenario where** developers from an insurance company must “keep the lights on” while evolving their containerized application – collecting relevant usage data and minimizing downtime.

**During the “hacking” attendees will focus on:**

1. Building a CI/CD pipeline from scratch that accommodates basic testing and deployment of cloud infrastructure and application.
2. Building out and improving the pipeline to implement security, monitoring, integration testing, and phased rollout.

**By the end of the OpenHack, attendees will have built out a technical solution** that is a complete development workflow using modern computing resources ([Azure App Service](https://docs.microsoft.com/en-us/azure/app-service/overview)).

**Technologies**

[GitHub](https://github.com/features) or [Azure DevOps](https://azure.microsoft.com/en-us/services/devops/) (*team choice*), [Azure App Service](https://docs.microsoft.com/en-us/azure/app-service/overview), [Log Analytics](https://docs.microsoft.com/en-us/azure/azure-monitor/logs/data-platform-logs), [Application Insights](https://docs.microsoft.com/en-us/azure/azure-monitor/app/app-insights-overview), [Azure Monitor](https://docs.microsoft.com/en-us/azure/azure-monitor/), [Azure SQL Database](https://docs.microsoft.com/en-us/azure/azure-sql/database/sql-database-paas-overview), [Azure Container Registry](https://docs.microsoft.com/en-us/azure/container-registry/container-registry-intro), [Key Vault](https://docs.microsoft.com/en-us/azure/key-vault/general/overview), [Bicep](https://docs.microsoft.com/en-us/azure/azure-resource-manager/bicep/overview), [Terraform](https://www.terraform.io/),

**Prerequisites**

**Knowledge Prerequisites**

To be successful and get the most out of this OpenHack, participants should have existing conceptual knowledge of DevOps best practices and advanced experience in Azure services, especially Azure App Service. This OpenHack is a 300-level learning experience, so it may be too complex if you are entirely new to Azure services. In addition, a fundamental experience of containers will also be helpful for this hack.

We recommend you read the following:

* [What is DevOps?](https://docs.microsoft.com/en-us/devops/what-is-devops)
* [DevOps: Bridging the gap between business and development](https://developer-tech.com/news/2016/jan/29/devops-bridging-gap-between-business-and-development/)
* [Introduction to Containers and Docker](https://docs.microsoft.com/en-us/dotnet/architecture/microservices/container-docker-introduction/)
* [What is Infrastructure as Code?](https://docs.microsoft.com/en-us/devops/deliver/what-is-infrastructure-as-code)
* [DevSecOps controls](https://docs.microsoft.com/en-us/azure/cloud-adoption-framework/secure/devsecops-controls)

Required knowledge of [Azure Fundamentals](https://docs.microsoft.com/en-us/learn/paths/az-900-describe-cloud-concepts/).

**Language-Specific Prerequisites**

* The sample application that is used throughout the event is composed of many APIs (Applications Programming Interfaces), each coded in a different language which offers coverage for teams who are proficient in several different programming languages.
* Hands-on coding is required in at least one of the following programing languages: C#, JavaScript/Node.JS, Golang or Java.

**Tooling Prerequisites**

To avoid any delays with downloading or installing tooling, have the following ready to go ahead of the OpenHack:

* A modern laptop running Windows 10 (21H2 or higher), Mac OS X (11.6 or higher), or Ubuntu (18.04 or higher)
* Install your choice of Integrated Development Environment (IDE) software, such as [Visual Studio Code](https://code.visualstudio.com/download).
* Download & install the latest version of: [Azure CLI](https://docs.microsoft.com/en-us/cli/azure/install-azure-cli?view=azure-cli-latest), [GitHub CLI](https://cli.github.com/), and [Terraform CLI](https://www.terraform.io/downloads.html).
* Optional install (it is only for someone who would like to test something on your local machine):
* [Docker for Windows](https://docs.docker.com/docker-for-windows/install/) or [Docker for Mac](https://docs.docker.com/docker-for-mac/install/)
* *Note:* If you are using Windows, you may want to enable Windows Subsystem for Linux and install Ubuntu or any other supporter distributions of Linux: <https://docs.microsoft.com/en-us/windows/wsl/install>

*Note:* Native Mac OS shell is not supported. You can still use Mac OS as your primary system, but you must make sure you have a Docker engine and run all OpenHack related stuff inside the interactive Docker container. Dockerfile with all required prerequisites is prepared and available in the OpenHack codebase.

**Development Environment Configuration**

None.

**Links & Resources**

Because you will be working in teams, a good overview of pair programming is useful. We recommend you read the following:

* [Pair Programming](https://www.agilealliance.org/glossary/pairing)

**Post Learning Recommendations**

* [GitHub learning paths](https://docs.microsoft.com/en-us/learn/browse/?resource_type=learning%20path&expanded=azure&products=github)
* [Azure DevOps learning paths](https://docs.microsoft.com/en-us/learn/browse/?resource_type=learning%20path&expanded=azure&products=azure-devops)

**Challenges**

**Challenge 1: Establish your plan**

To have a successful DevOps strategy, your team needs to have a plan. In this challenge, your team will learn about the basics of DevOps and gain an understanding of a team development model.

Learning objectives:

* Adopt a DevOps “mindset” in the team.
* The participants are asked to get to know themselves better, organize the team and define how and where they will handle work items.
* Design task board to track work in progress.

**Challenge 2: Setting up the Development Workflow**

In this challenge, teams will select tools to meet the planning objectives as it relates to continuous integration.

Learning objectives:

* The participants will learn the fundamentals of Planning and Continuous Integration using the tooling of their choice.
* By using Branch Protection, teams can enforce code quality policies for incoming changes ensuring that each service has a code owner.
* Participants review pull requests before changes are merged into the main code base.

**Challenge 3: Deploy Cloud Infrastructure with Infrastructure as Code (IaC)**

In this challenge, teams will build workflow for cloud infrastructure deployment.

Learning objectives:

* Participants will learn how to deploy cloud infrastructure on Azure using infrastructure definition from the code.

**Challenge 4: Implement Continuous Integration (CI) with Testing**

In this challenge, teams will improve the reliability of their code by introducing testing.

Learning objectives:

* Participants will learn to integrate unit tests into a build workflow to supply rapid feedback and augment thorough code reviews designed to only accept changes into main when all tests pass.
* They will be asked to run unit tests automatically and integrate them in the workflow they are building.

**Challenge 5: Implement Continuous Deployment (CD)**

In this challenge, teams will focus on release management automation.

Learning objectives:

* The participants will learn the fundamentals of Release Management by automatically deploying an updated version of their application to an Azure App Service.
* They will also be asked to demonstrate that the respective container images are only updated when changes are successfully merged into the main branch.

**Challenge 6: Implement a Blue/Green deployment strategy**

In this challenge, teams will learn how to perform a zero down time deployment using a Blue/Green deployment strategy.

Learning objectives:

* Building on earlier challenges, participants will learn how to implement a blue/green deployment strategy.
* They will be asked to articulate the blue/green logic and demonstrate its implementation for one of the APIs of the provided application, so that they are able roll out code changes without causing application downtime.

**Challenge 7: DevSecOps – Get rid of secrets**

In this challenge, teams will learn about securing codebase and prevent leaks of secrets.

Learning objectives:

* The participants will learn how to detect secrets in the codebase and move all sensitive data to secure place.
* They will be asked to implement a secret rotation strategy for SQL database.

**Challenge 8: Integrating quality and security gates**

In this challenge, teams will learn how to enhance their testing strategy through integration with external tools focused on improving the quality of the software that is released into production.

Learning objectives:

* The participants will be required to improve the automated testing capabilities of their pipelines to incorporate more sophisticated quality and security checks.
* They will also be required to demonstrate one or more of the following enhancements to their pipeline: Dependency scanning, SAST, DAST, Variant Analysis, Code Coverage, Integration Tests, Load Tests and Manual Approval prior to deployment.

**Challenge 9: Implement a monitoring solution with alerting**

In this challenge, teams will learn about configuring a monitoring and alerting solution.

Learning objectives:

* The participants will learn how to close the DevOps loop by adding monitoring and alerting.
* They will be asked to demonstrate a view aggregating the monitoring of the application and infrastructure. They also will have to implement alerting in the case of application performance degradation. Alerts should also generate a work item in the teamwork tracking system.

**Challenge 10: Implement phased rollout with rollback**

In this challenge, teams will learn how to enhance their Blue/Green Deployment strategy by performing a gradual release of software into production with a rollback strategy.

Learning objectives:

* The participants will revise their deployment strategy and learn how to perform a blue/green deployment with gradual rollout and how to implement a “rollback” mechanism.
* They will be asked to add several phases to their existing pipeline to support a gradual rollout/rollback of a new version of the application. They will be asked to implement gates to validate the behavior of the application and implement a rollback mechanism.

**Challenges flow**

Diagram

Description automatically generated

**Value Proposition**

DevOps fundamental upskilling and developing secure zero-downtime deployment strategies, translates to reduced friction in production deployments and ensures deployments of new features can occur more often and safely without requiring system downtime.

**Technical Scenarios**

* "Keeping the lights on” – implementing a production pipeline that alleviates the problem of high downtime when making new development changes to your application
* Testing – Unit, Integration and Load testing to reduce the risk of “breaking production” and ensuring that new code will integrate properly with current code
* Phased rollout – gradual code change implementation and ability to “rollback” to add a layer of security to production.
* Security – implementing DevSecOps practices to reduce risk of secrets leaks and improve overall code and application security.

**Audience**

* Target Audience:
  + Microsoft – CSE, CSA, GBB, ATT, SE, TPM
  + Customer – App Developers
* Target verticals: Cross-Industry
* Customer profile:
  + Enterprises and ISVs looking to deploy containerized based workloads in the cloud
  + Customers looking forward to using containers for DevOps
  + Customers looking forward to micro-services architecture for their existing or new solutions
  + Customers looking to deepen their overall DevOps maturity

**Registration Questions**

|  |  |  |
| --- | --- | --- |
| **Required** | **Field** | **Response Options** |
| Yes | What is your level of understanding using Azure Cloud today? | * None * Some understanding * I have some pilot work on Azure * I rely on Azure today for cloud |
| Yes | How much time have you spent with GitHub? | [Dropdown or Radio Buttons]   * None * < 3 months * 3-6 months * 6 months - 1 year * > 1 year |
| Yes | How much time have you spent with Azure DevOps? | [Dropdown or Radio Buttons]   * None * < 3 months * 3-6 months * 6 months - 1 year * > 1 year |
| Yes | How much time have you spent with Infrastructure as Code practices, such as Terraform, Bicep, ARM? | [Dropdown or Radio Buttons]   * None * < 3 months * 3-6 months * 6 months - 1 year * > 1 year |
| Yes | How much time have you spent with DevOps practices, such as CI/CD? | [Dropdown or Radio Buttons]   * None * < 3 months * 3-6 months * 6 months - 1 year * > 1 year |
| Yes | How much time have you spent with DevSecOps practices, such as Static/Dynamic Code Analysis, Dependency Scanning? | [Dropdown or Radio Buttons]   * None * < 3 months * 3-6 months * 6 months - 1 year * > 1 year |
| Yes | What is your preferred learning path during the event? | [Dropdown or Radio Buttons]   * GitHub * Azure DevOps |

**Coach sourcing questions**

The following questions can be used to source coaches with appropriate knowledge. See Coach Selection in the [OpenHack Playbook](https://aka.ms/openhackplaybook) for additional information.

|  |  |  |
| --- | --- | --- |
| **Question #** | **Question** | **What to look for in answer** |
| 1 | Have you used open-source CI/CD in a project to build and release software? | Should answer yes. |
| 2 | Have you worked on a project that involved deploying microservices to Azure Cloud in the last 12 months? | Should answer yes. |
| 3 | Have you deployed a web app using deployment slots in an app service using GitHub or Azure DevOps? | Should answer yes. |
| 4 | Have you led a project that used agile or scrum practices with CI/CD and Git version control? | Should answer yes. |
| 5 | Have you worked on a project that involved deploying cloud infrastructure using Infrastructure as Code practices (such as Terraform, Bicep, or ARM) in the last 12 months? | Should answer yes. |
| 6 | Have you used any DevSecOps practices to improve security of codebase and application in the last 12 months? | Should answer yes. |

**FAQ**

Q: Is there a suggested flow of OpenHacks that an attendee should attend first before attending?

A: No, but previous knowledge of Containers will help attendees.

Q: If I am only interested in using FaaS for compute, should I attend this OpenHack?

A: No, Serverless services are not covered in this OpenHack – only Containers will be addressed.