**TECHSTYLERS DEEP DIVE - COHORT 6.0**

# CLOUD COMPUTING WITH AZURE

# PROJECT REPORT

**Team:** TEAM CHARIS  
**Project Title:** Automating Infrastructure Deployment **with IaC (Bicep & JSON Templates)**  
**Duration:** June 7 – June 27, 2025

### **Introduction**

This report documents the complete lifecycle of the Automating Infrastructure Deployment project carried out by TEAM CHARIS as part of the TechStylers Deep Dive Cohort 6.0. The goal of the project was to automate the deployment of a secure and scalable Azure infrastructure using Infrastructure as Code (IaC), specifically leveraging **Bicep** and **ARM (JSON) templates**.

This initiative aimed to eliminate manual provisioning errors, enhance deployment speed, and enforce governance, security, and monitoring policies across the infrastructure environment.

### **Team Structure and Roles**

|  |  |  |
| --- | --- | --- |
| Name | Role | Responsibilities |
| Aishat Oshileye | Project Manager | Lead planning, track progress, facilitate communication, and conduct reviews |
| Rebecca Oburotha | Identity/Security Lead | Implement PIM, manage role access, create and assign Azure Policies |
| Victor Nwoke | VM/Networking Lead | Deploy VMs, set up VNets, configure subnets and NSGs |
| Lucy Obilor/ Oluwatobiloba Aladetuyi | Storage Lead | Automate storage accounts, configure tiers and access control |
| John Oche | Monitoring Lead | Set up Azure Monitor, define alerts, configure diagnostic logs and action groups |
| Oladimeji Williams | Documentation Lead | Maintain proposal, report, implementation guides, and user documentation |
| Miracle Clement | Presentation Lead | Develop slide decks, organize the demo, summarize project highlights and challenges |

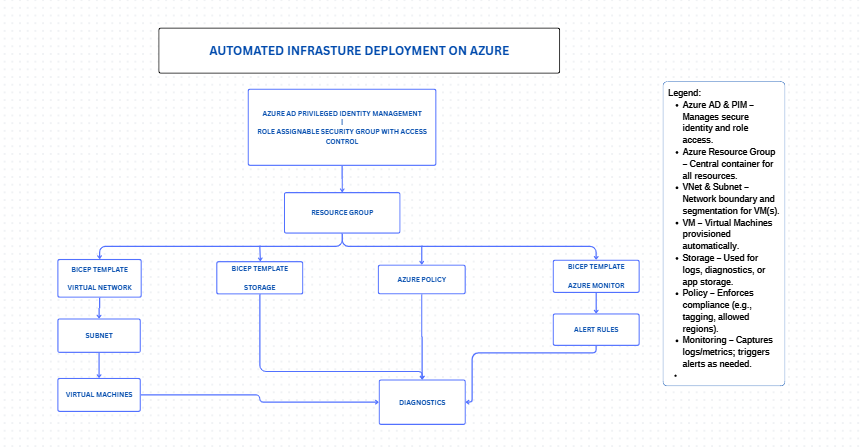
### **Architecture Overview**

The design centered on a modular infrastructure, where each service was deployed through a reusable Bicep or JSON template.

**Key Components:**

* **Azure AD PIM**: Managed privileged access using JIT role activation and MFA.
* **Virtual Network & NSGs**: Defined segmented subnets and security rules.
* **Virtual Machines**: Deployed Linux and Windows VMs with parameterized templates.
* **Storage Accounts**: Provisioned with specific SKUs, hot access tiers, and diagnostic settings.
* **Azure Policy**: Enforced tagging, region restrictions, encryption audits.
* **Azure Monitor**: Monitored VM metrics, logged diagnostics, and issued alerts.

#### **Architectural Diagram**



## Benefits/Business Value

**Accelerated Time-to-Market**

* Spin up complete environments in minutes, not days.
* Rapid prototyping, testing, and delivery of new features.

**Consistency & Repeatability**

* Infrastructure-as-Code (IaC) ensures every deployment is identical.
* Eliminates “it works on my machine” issues across dev/test/prod.

**Reduced Human Error**

* Automation minimizes manual steps, avoiding misconfigurations and downtime.

**Cost Optimization**

* Automate resource tagging for chargeback or cost analysis.
* Enforce policies to control overspending (e.g., limit VM SKUs).

**Enhanced Security & Compliance**

* Automatically apply role-based access, encryption, and network rules.
* Enforce organizational standards through Azure Policy.

**Improved Scalability**

* Rapidly replicate infrastructure across regions or projects.
* Meet customer demand without infrastructure bottlenecks.

**Greater Developer Productivity**

* Developers focus on coding, not setting up infrastructure.
* Self-service environments via automation empower faster innovation.

**Traceability & Accountability**

* Tags and policies create a clear audit trail of resource ownership and purpose.
* Simplifies troubleshooting and change tracking.

**Resilience & Observability**

* Automated integration with Azure Monitor ensures visibility into system health.
* Quick detection and response to performance or security incidents.

**Competitive Advantage**

* Faster releases, better reliability, and lower costs help deliver better products ahead of competitors.

### **Implementation Details**

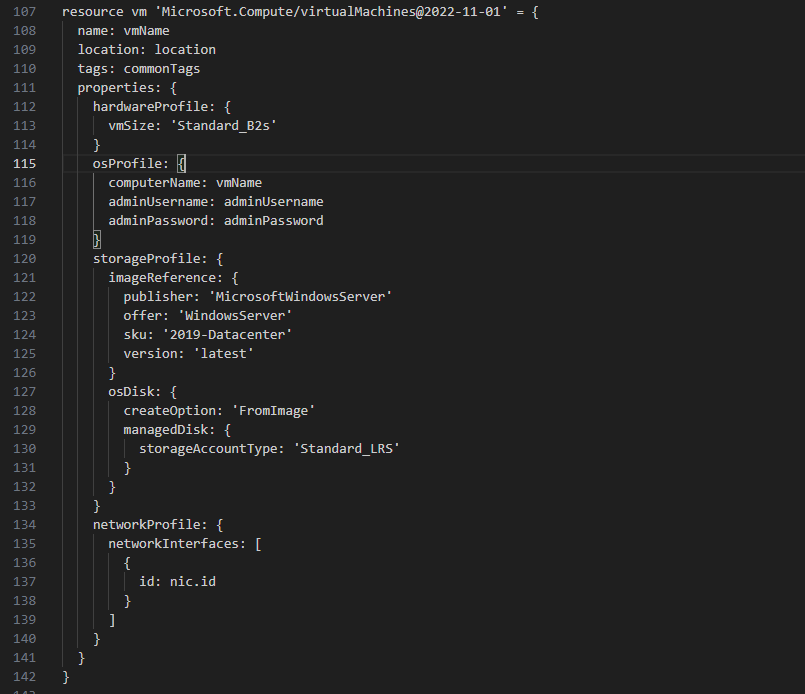
#### **a. Identity (Azure AD PIM)**

* Assigned eligible roles with approval workflows and MFA requirements.
* Validated with test users and reviewed audit logs.

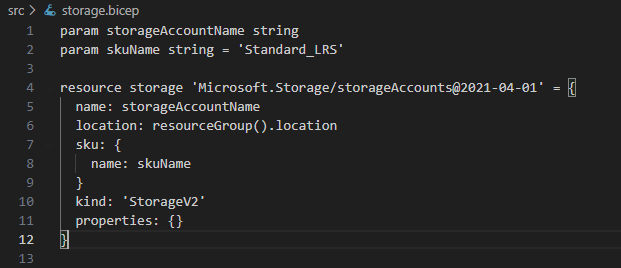
#### **b. Networking (Bicep)**

* VNet with address space 10.0.0.0/16
* Subnets per component (VM, Database)
* NSGs with inbound rules for RDP/SSH only

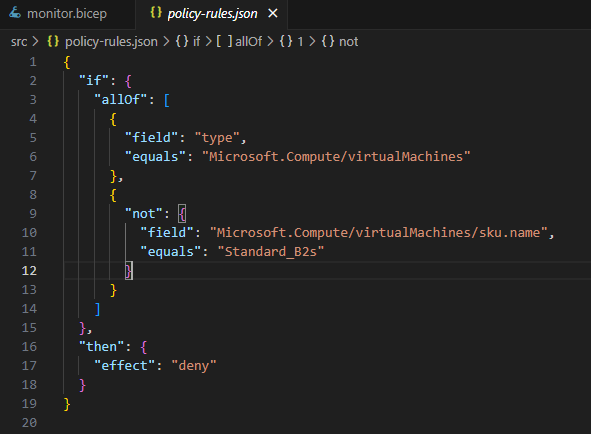
#### **bi. Virtual Machines (Bicep)**



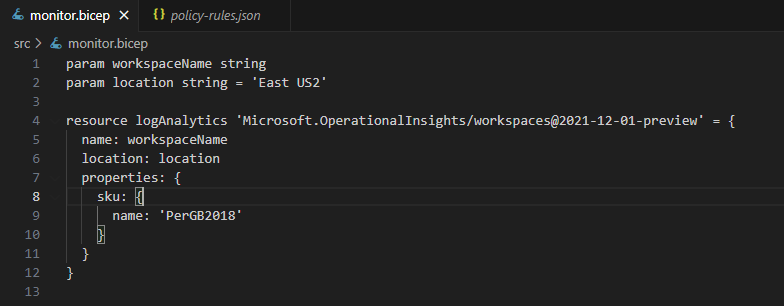
#### **c. Storage (Bicep)**



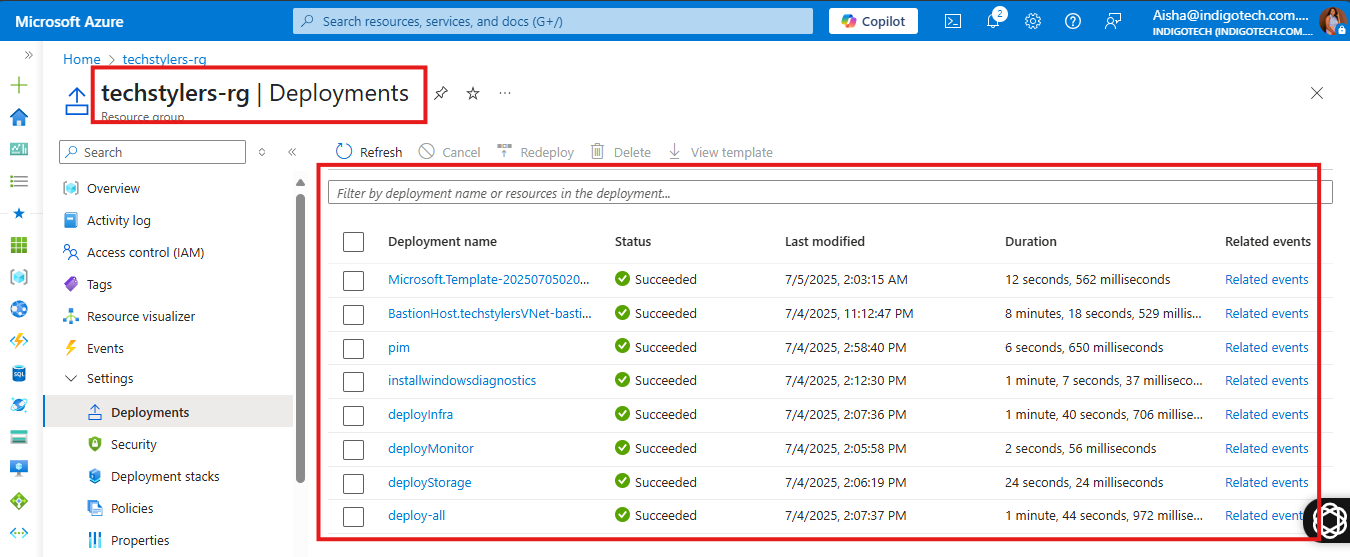
#### **d. Security (Azure Policy - JSON)**



#### **e. Monitoring (Bicep)**

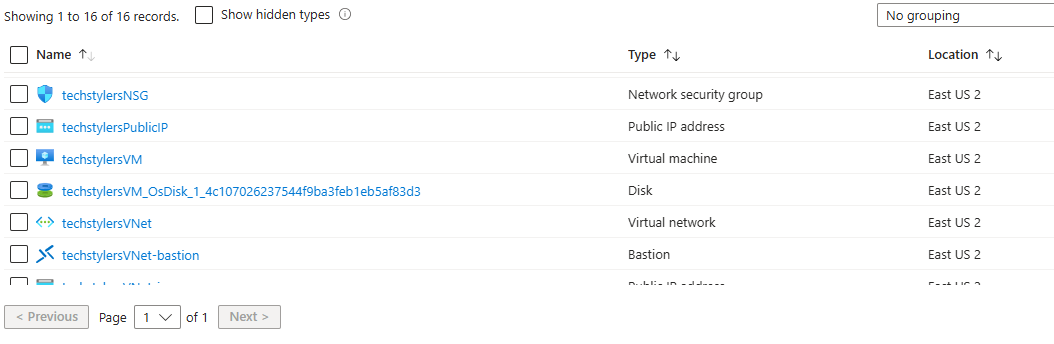


### Environment Screenshots



A screenshot of a computer

AI-generated content may be incorrect.



An overview of deployments showcasing deployment speed

### **Tools Used**

* **Azure Portal** – Manual validation and testing
* **Azure CLI & PowerShell** – Deploying and validating templates
* **Visual Studio Code** – Authoring Bicep & JSON templates
* **GitHub** – Hosting and tracking version control
* **Microsoft Learn** – Referenced for role-based research

### **Testing & Evaluation**

Each template was validated using:

az deployment group validate --resource-group ‘techstylers-rg’ --template-file deploy-all.bicep

* VM deployments were verified via IP access.
* Policy effectiveness was tested by attempting non-compliant deployments.
* Alerts were simulated using CPU-intensive scripts.

## Challenges Encountered

1. **Subcription constraint:**

To implement this project, the team had to set up a Azure trial subscription tenant. This subscription however had some limitations as it lacked some licences needed to implement some solutions which was not available within the free trial Azure subscription we used.

1. **Automated Identity provisioning And Access Control**

To automate identity management, we tried using Powershell, which required us to install the Microsoft Graph module, we encountered some issues during these processes, until we discovered that we were constrained by the lack of the appropriate license (for PIM). We later found a walk-around where we merged the Azure subscription with a Microsoft 365 Developer tenant to use the PIM feature.

### **Lessons Learned**

* **Parameterization** makes templates reusable and adaptable.
* **Debugging Bicep** requires careful attention to resource names and types.
* **Monitoring setup** must align with business-critical metrics.
* **PIM access controls** enhance security but requires user onboarding.

### **Conclusion**

This project enabled our team to implement a real-world Azure automation solution using modern cloud practices. It emphasized teamwork, practical learning, and secure automation. The modular Bicep and JSON template structure laid a strong foundation for scalable infrastructure.

We are proud of the results and the knowledge gained.

You can find all the files related to this project in the github repository below:  
<https://github.com/Oladimeji-Williams/Cloud-Computing-With-Azure>

**Prepared by:**  
TEAM CHARIS – TechStylers Tech Deep-Dive Cohort 6.0