**TECHSTYLERS DEEP DIVE COHORT 6.0**

**PROJECT TEAM CHARIS**

**PROJECT PROPOSAL**

# Automating Infrastructure Deployment

## Executive Summary

This proposal presents a smart and secure way to set up and manage cloud resources in Microsoft Azure using automation. The system is designed for one organization (single tenant) and follows strong best practices for organizing, protecting, and maintaining cloud services.

Everything is built using code (called “Infrastructure as Code”) with tools like **Bicep** and **PowerShell**. This lets the team deploy virtual machines, storage, networking, and monitoring tools quickly, consistently, and with fewer mistakes. All the work can be done with scripts—no need to click around in the Azure portal.

To manage who can access what, the solution uses **Azure AD** along with **Privileged Identity Management (PIM)**. This gives temporary, secure access to important roles, and only when it’s needed. It also allows us to create and manage user accounts through automation using Microsoft Graph.

The resources are grouped by project environments—like development or production—so it’s easy to manage, secure, and scale. The network uses a structure called **hub-and-spoke**, which keeps systems connected but separated for better control and security.

Security is a top priority. **Azure Policy** is used to make sure everything follows company rules (like requiring secure data connections), and **Azure Key Vault** stores secret passwords safely. Tools like **Microsoft Defender for Cloud** help watch for any threats.

All systems are monitored using **Azure Monitor** and **Log Analytics**, which let the team track performance, catch problems early, and set alerts when something needs attention.

Lastly, this setup can be connected to **CI/CD pipelines** using tools like GitHub Actions. That means every time the team makes a change to the code, the system can automatically deploy it, saving time and improving reliability.

This cloud setup is secure, efficient, and ready to grow or scale with the needs of the organization.

The project spans planning, implementation, delivery, and appraisal stages to ensure holistic and successful learning and technical experience for all team members.

## Project Objectives

This proposal presents a tailored solution which aims:

* To deploy a fully automated infrastructure solution that supports scalability on Microsoft Azure.
* To reduce manual errors and increase consistency in deployment workflows.
* To enable governance, security, and monitoring of cloud resources.
* To upskill team members in cloud automation tools and methodologies.

## Problem Statement

The current approach to provisioning cloud resources is manual, making it difficult to efficiently support multiple projects with standardized configurations. This method introduces inconsistencies, increases the likelihood of errors, and reduces both the speed and quality of deployments, ultimately impacting business value.

As a software development company, our client requires an Azure infrastructure that is automated, secure, and scalable.

Relying on manual resource creation and configuration not only extends deployment timelines but also increases operational risks, resulting in an unreliable and less repeatable infrastructure setup.

## Scope of Work

The project will cover the deployment of Azure resources using Bicep/ARM templates, an Infrastructure as Code (IaC) solution that automates Azure resource deployment.

Resource Manager provides tools to support infrastructure as code (IaC). This project defines IaC for all solutions in the cloud, but it's especially valuable for multitenant solutions.

The solution uses Bicep to deploy IaC from a deployment pipeline. Bicep is a language designed to deploy and manage Azure resources in a declarative way. You can also use JSON Azure Resource Manager templates (ARM templates), Terraform, or other partner products that access the underlying Resource Manager APIs. This project is scoped to automate the provisioning of the following resources using Bicep templates:

* Microsoft Entra ID Privileged Identity Management (PIM)
* Virtual Machines (Windows/Linux)
* Azure Storage Accounts
* Virtual Networks (VNets) and Network Security Groups (NSGs)
* Azure Policy (Governance)
* Azure Monitor (Alerts & Logs)

## Project Planning

**Project Phases:**

|  |  |  |
| --- | --- | --- |
| Phase | Timeline | Activities |
| Planning | June 7 – June 9 | Define scope, assign roles, distribute learning resources, schedule kickoff meeting |
| Development | June 10 – June 20 | Design Bicep templates, implement infrastructure modules, set up GitHub repository |
| Testing | June 21 – June 24 | Deploy in sandbox, validate templates, simulate alerts, verify policy compliance |
| Delivery | June 25 – June 26 | Prepare final report, presentation slides, GitHub README, conduct demo rehearsal |
| Appraisal | June 27 | Deliver live demo, presentation, conduct peer/team evaluations, and collect feedback |

## **Team Structure & Roles**

At Team Charis, we boast of a vibrant and innovative team of professionals with in the provisioning and management services of cloud solutions. Meet the Team Charis:

|  |  |  |
| --- | --- | --- |
| 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 |
| 1. Lucy Obilor 2. 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 |

## **Tenancy and Architectural Considerations**

**1. Tenancy Model: Single Tenant with Management Group Hierarchy**

The solution is best implemented within a **single Azure Active Directory (Azure AD) tenant**, which provides centralized identity management, unified security policies, and consistent role governance.

This approach is ideal for organizations managing resources under a common identity platform while adhering to compliance and operational standards.

To enhance governance and scalability, the deployment adopts an **Azure Management Group hierarchy**, structured as follows:

* **Root Management Group**
  + **Subscription 1**
* **Development Projects – Management Group**
  + **techstylersdev-rg – Resource Group**
* **Production Projects – Management Group**
  + **techstylers-rg – Resource Group**
* **Future initiatives - Subscription 2**

This hierarchy enables the inheritance of policies, budgets, and access control, reducing administrative overhead and supporting long-term organizational growth.

**2. Architectural Design**

The architectural design of the automated Azure infrastructure solution ensures modularity, security, scalability, and continuous monitoring. Below is a breakdown of the core components:

a. Identity and Access Management

* Azure Active Directory (Azure AD) provides the foundation for access control.
* Privileged Identity Management (PIM) manages just-in-time admin role assignments with MFA.

b. Compute Layer (Virtual Machines)

* One or more VMs are deployed in a designated subnet.
* Each VM is configured using a Bicep module for consistent specifications, including size, image, and tags.

c. Networking Layer

* A Virtual Network (VNet) spans the deployment region with segmented subnets.
* NSGs control traffic in and out of subnets.
* Public IPs are assigned conditionally, with DNS labels as needed.

d. Storage Layer

* Azure Storage Accounts are provisioned for blob, file, and table services.
* Storage settings enforce encryption, access tiers, firewalls, and logging.

e. Security and Governance

* Azure Policies enforce rules on tag compliance, allowed regions, and storage encryption.
* Role-Based Access Control (RBAC) limits access to specific scopes.

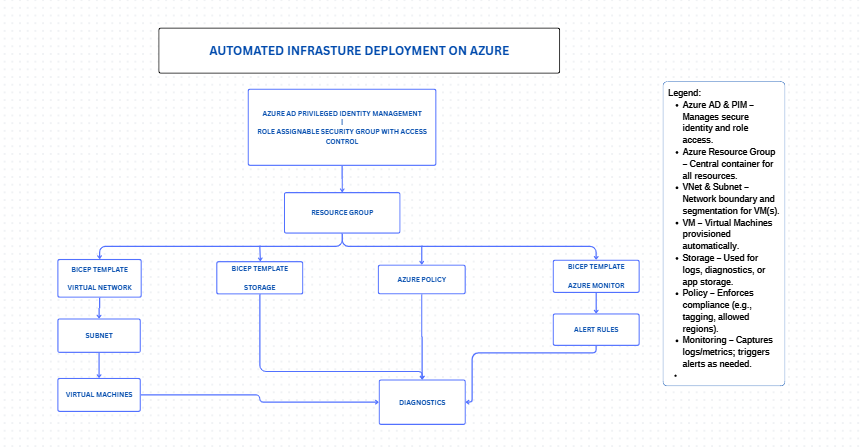
f. Monitoring and Observability

* Azure Monitor collects logs and metrics from VMs and storage.
* Alerts are configured with thresholds and tied to Action Groups for notifications.

g. GitHub Repository and CI/CD (Optional)

* Templates and scripts are stored on GitHub.
* GitHub Actions may trigger deployments upon code pushes.

### **ARCHITECTURAL DIAGRAM**



## **Tools and Technologies**

* **Microsoft Azure** – Cloud provider platform
* **Azure CLI** – Command-line management for running and executing commands
* **Bicep** – IaC language for resource deployment
* **ARM Templates** – Declarative JSON-based resource definitions
* **Azure Portal** – For real time Manual verification and monitoring
* **GitHub** – For Version control, documentation and collaboration
* **VS Code** – Development environment
* **Azure Monitor** – Logging and alerting for proper resources monitoring
* **Azure Policy** – Governance and compliance enforcement

## **Strategies for Success**

* **Modular Template Design**: Templates will be split into reusable modules (e.g., VM, VNet, Storage, Monitor) to promote maintainability.
* **Parameterization**: All templates will be parameterized for flexibility across environments.
* **Validation Pipeline**: Scripts and templates will be validated using Azure CLI before deployment.
* **Sandbox Testing**: All deployments will be tested in non-production environments.
* **Collaborative Learning**: Team members will share resources, code reviews, and troubleshooting sessions.
* **Weekly Check-ins**: To ensure accountability, track progress, and adjust timelines as needed.
* **CI/CD Integration** *(Optional)*: GitHub repository will be used to store templates and scripts. GitHub Actions may be used for continuous deployment of workflows.

## **Implementation Plan**

**a. Identity and Access Management**

* Setup PIM to limit admin access against role-assigned security group
* Configure role eligibility, MFA, and activation settings
* Apply settings to role groups

**b. Virtual Machine Deployment**

* Create Bicep templates for VM provisioning
* Include custom parameters: name, size, region, OS, tags

**c. Storage Automation**

* Bicep templates for Storage Accounts
* Configure SKU, redundancy, tiers, firewall rules

**d. Networking Setup**

* Deploy VNets and subnets with NSGs
* Use isolated subnets for security zones

**e. Security & Compliance**

* Create and assign Azure Policies (tagging, allowed locations)
* Validate compliance through Azure dashboard

**f. Monitoring and Alerting**

* Configure metrics (e.g., CPU, Disk usage)
* Set up email alerts via Action Groups

**g. Deploy all templates**

## Delivery Artifacts

* Final Presentation Deck
* Live/Recorded Demo
* GitHub Repository with:
  + /templates – All Bicep/ARM code
  + /monitoring – Alert definitions
  + /docs – Guides, diagrams, and notes
  + README.md – Setup instructions and project overview
* Project Report
* Feedback/Evaluation Summary

## Conclusion

This project will provide hands-on, real-world exposure to cloud infrastructure deployment using automation and modern DevOps practices. With a strong team, clear planning, and the right tools, we are confident this project will be both impactful and rewarding.

**Prepared by:**  
Cohort 6 – Team Charis

For:

**Techstylers Deep Dive**