ITC INFOTECH

Business-friendly Solutions

ITC Infotech
Cloud Native Solution Framework

APRIL 2022

Topics

01 ITC Infotech Cloud Native Offering

02 ITC Infotech Migration Strategy

03 Cloud Native Solution Framework

04 Cloud Native – Modernization Themes
- Application Containerization
- Application Re-factoring
- API First Methodology
- DB Modernization
- DevOps Modernization

05 Common Solution Patterns
- Legacy to Modernization
- Cross Cutting –Common Concerns

06 Tools & Accelerators

07 Governance

ITC Infotech Cloud Native Offerings

ITC Infotech - Service Offerings and Frameworks

Strategy & Consulting

- Cloud Native application strategy, planning and implementation approach

Maturity Assessment & Roadmap

- Cloud and Cloud native capabilities assessment
- Adoption Plan, Reference architecture blueprints

Migration Services

- Application migration on Cloud native capabilities
- Enhance with Cloud containerization and DevOps

Engineering

- Application Modernization with cloud native technologies
- API and Microservices based design and development on cloud
- Serverless architecture services development

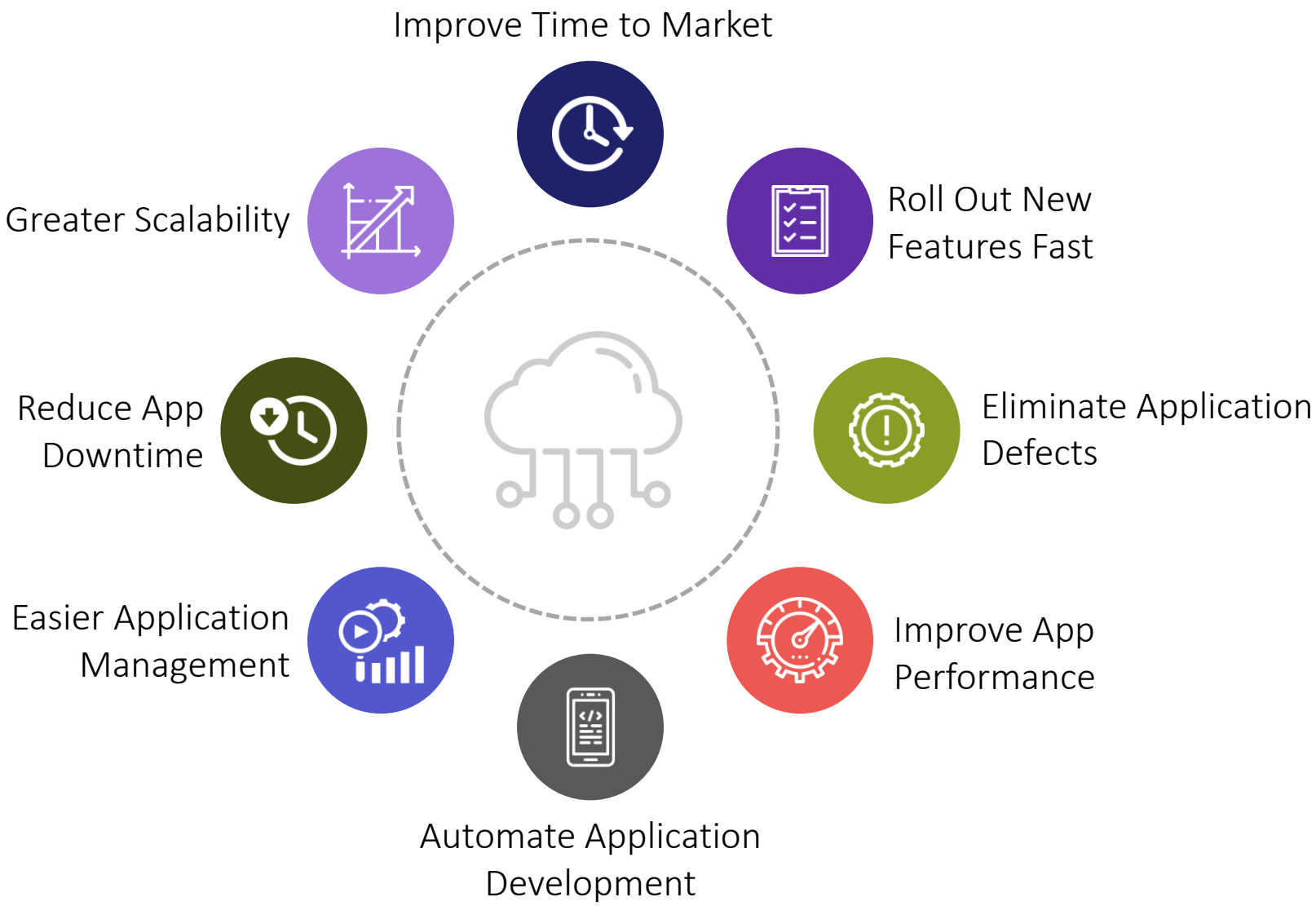
Technology

- Azure Cloud native experience
- Large Scale Implementation experience
- API Led Digital transformation and Digital Integration Experience

Value Adds & Accelerators

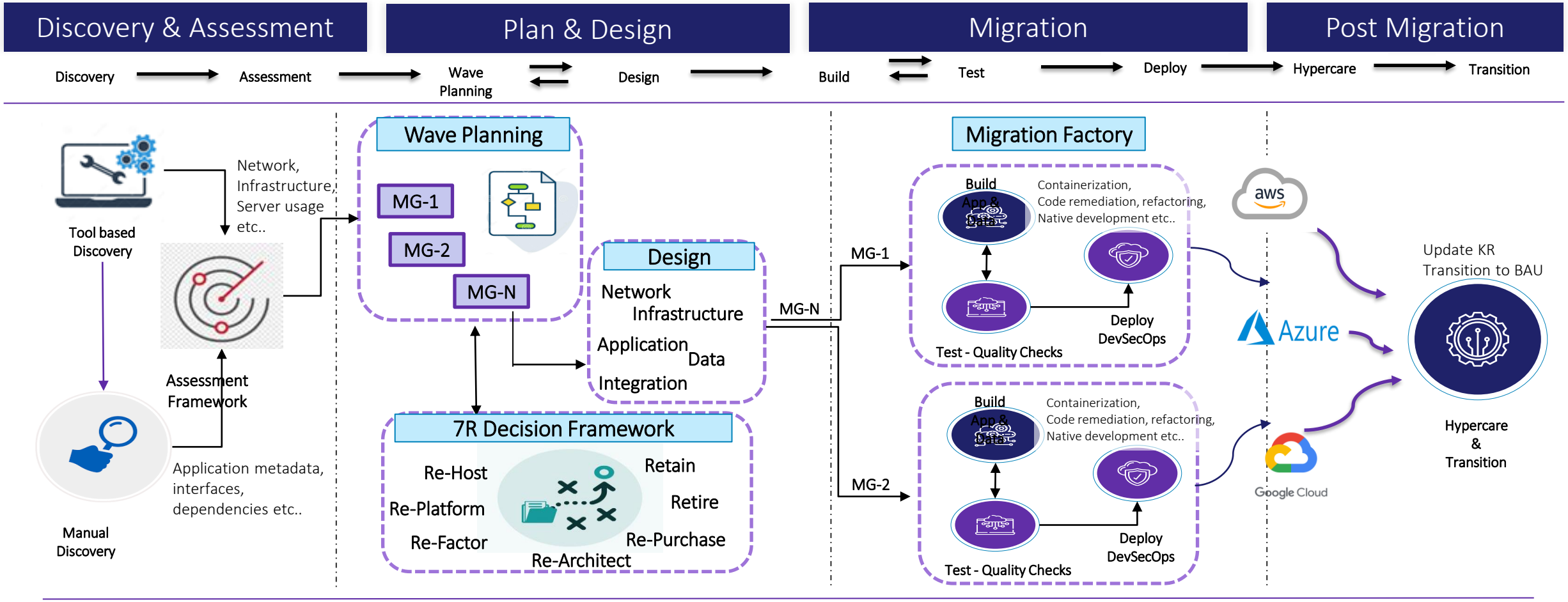
- Implementation Accelerators reduce overall Implementation Effort
  - Microservices Assessment framework
  - Integrated Development & Deployment framework

Our Cloud Native Solution - Benefits



ITC Infotech Migration Strategy

Migration Methodology - Phased Approach



Project Steering and Program Governance

- Program Organization – ITC, Customer, & RACI
- Program Governance
- Customer Engagement & Priority identifications
- Proactive Risk Identification & Mitigation
- SLA & KPIs
- Command Center & Status updates
- Change Management

Our Cloud Native Solution Framework

Cloud Native Solution Framework

Governance

Business Outcome

Business strategy, Goals and Runbook

Standards & Policies

Implementation guidelines and standards to establish cloud native capabilities

Process

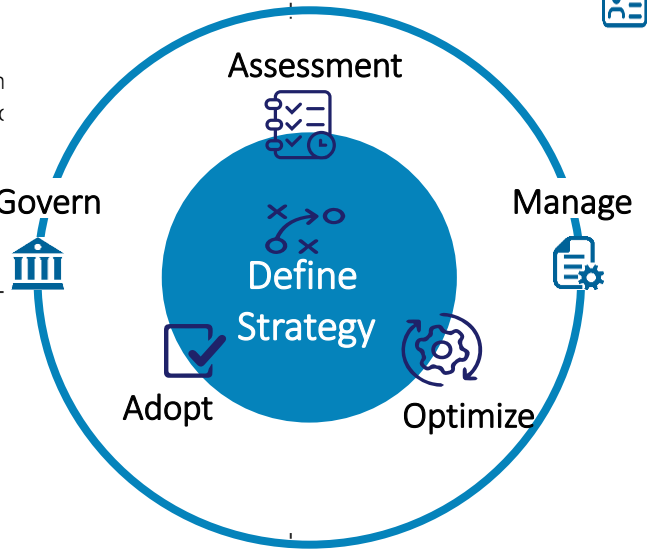
Establish processes to monitor violation and adherence to policies

- Process
- Technology
- Asset
- People

Strategy and Assessment Approach

BUSINESS → Microservice → TECHNOLOGY, PROCESS AND PEOPLE → Agile → DevSecOps → Cloud

- Business Events/ Requirements
- Architecture Style
- Development Process
- Automation Process
- Execution Platform



Manage

Criticality

Document the criticality and relative business value of each service

Impact

Establish clear performance expectations and business KPIs

Business Value

Document, track and report on commitments to cost and performance

- Operational Efficiency
- Monitoring
- Platform Operations

Adopt and Optimize

Build

Iterative build model to meet defined customer need /functional behavior

Review

Establish repeatable processes for iterative review for design betterment

Optimize

Gather metrics and optimize services and processes

- Containerized Architecture
- API & Microservice
- Serverless Architecture
- DevSecOps

Cloud Native - Modernization Themes

Modernization Themes – Application Containerization

Evaluate Enterprise Legacy (Monolith) Inventory Applications

Gather Application Requirements

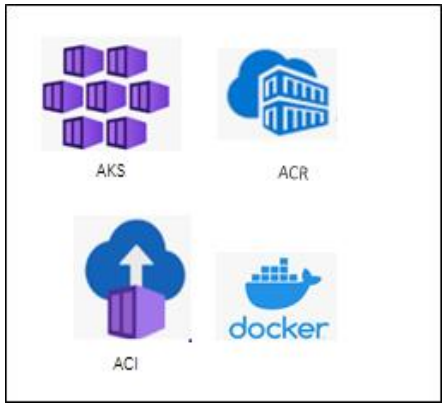
- Analyze and gather all the system requirements required for Containerization

- Authentication
- Security
- Architecture
- Performance
- High Availability
- Monitoring
- Scalability
- Re-usability
- Downtime
- Database
- Filesystems

Application Containerization

- Identify applications
- Choose a base Image.
- Install the necessary packages.
- Add your custom files.
- Define which user will (or can) run your container.
- Define the exposed ports.
- Define the entry point.
- Define a Configuration method.
- Externalize your data.
- Set up container Security & Governance

Tools & Services



Container Infrastructure Planning

- Identity key Infrastructure components required for converting the monolith application into a containerized application

Containerization of Applications

- Set up Containerization for identified and defined applications in the portfolio

CI-CD / DevOps Integration

- Set up CICD / DevOps Integration for the applications

Container Governance

Container Security

- Set up Container Security

Test & Rollout to cloud

- Test and Rollout application in Phases

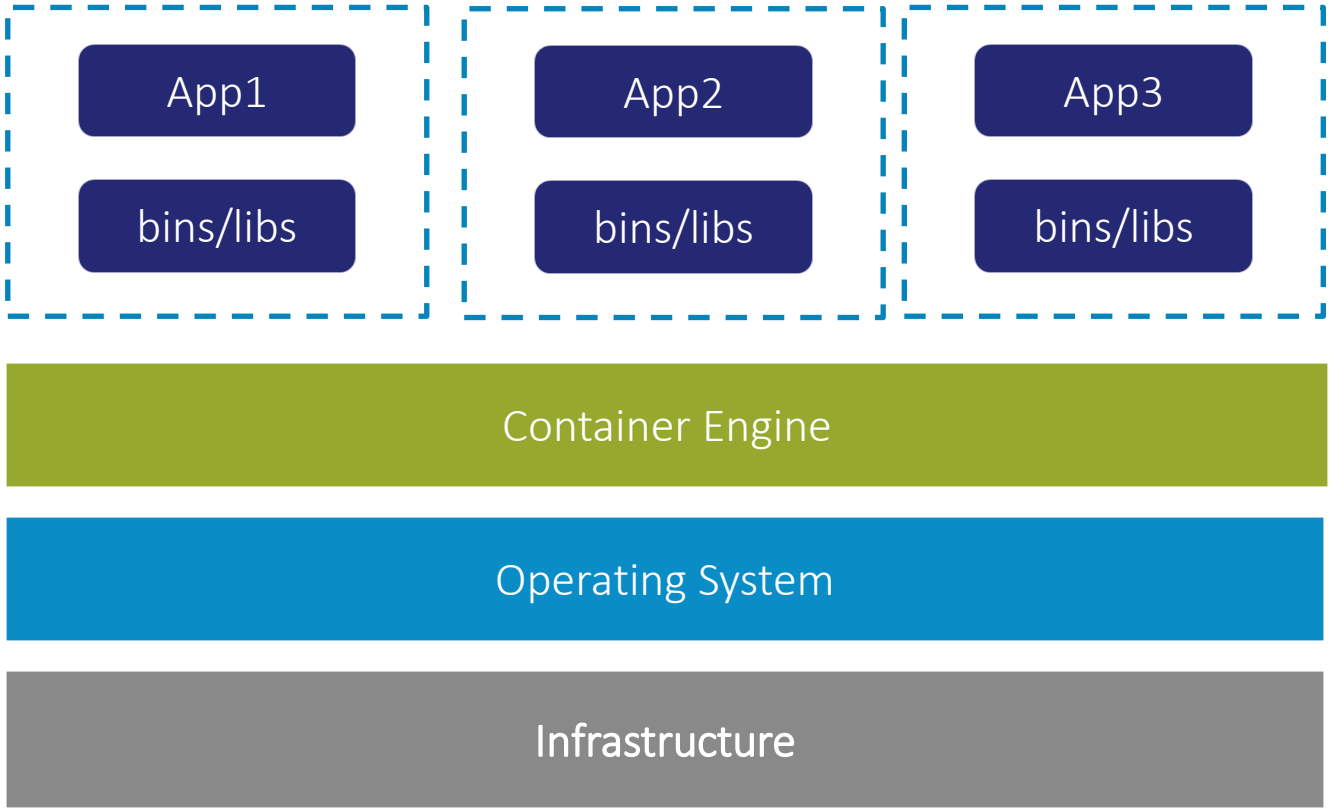
Performance & Monitoring

- Identity key performance metrics

Feedback & Optimizations

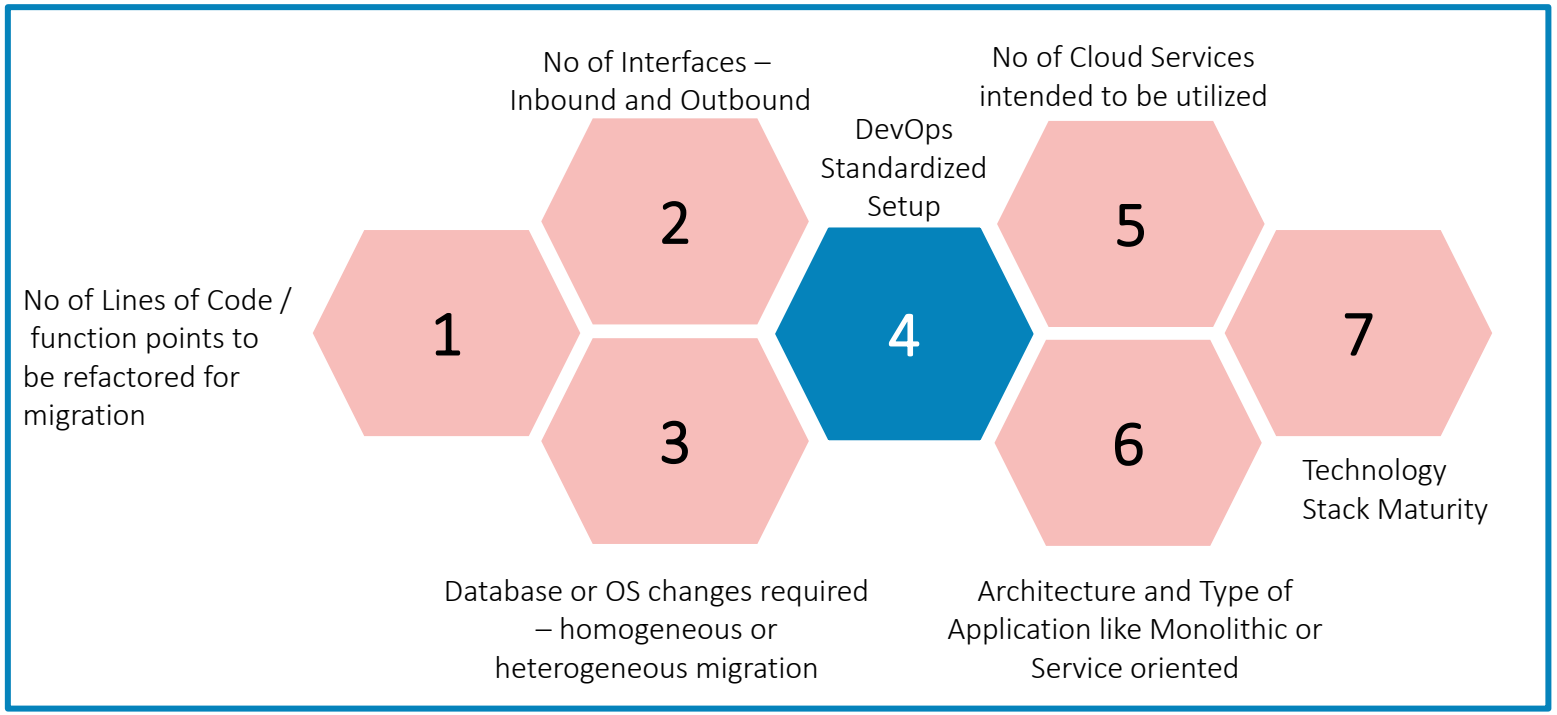
- Review user's feedback and overall application usability and performance and optimize as an when required.

Modernization Themes – Application Containerization Contd.



- Package once , deploy everywhere with less risk.
- Portability
- Supports Infrastructure-as-a-Code principles
- DEV/PROD environment parity from desktop to production environments
- Better community support including Docker
- Supports scalable workload
- Provides application and process isolation
- Better Performance
- Ease of deployment
- More control on the technology stack
- Prefer more serverless architecture over infrastructure

Modernization Themes – Application Refactoring



Application Refactoring

- Identify key enterprise apps
- Define Application parameters like authentication , security , performance etc.
- Identify the lines of code and touch points for refactoring
- Define no of Interfaces – Inbound and Outbound
- Setup CICD process
- Define well defined Architecture
- Leverage serverless cloud platform
- Continuous Health Monitoring
- Automated Alerts and Notifications

Tools & Services

Considerations

Application Architecture
- No Existing Webservices
- Existing Macro Services
- REST or JMS
- SOAP or EJB
- Servlet / JSP
- JSF / Spring Portlet

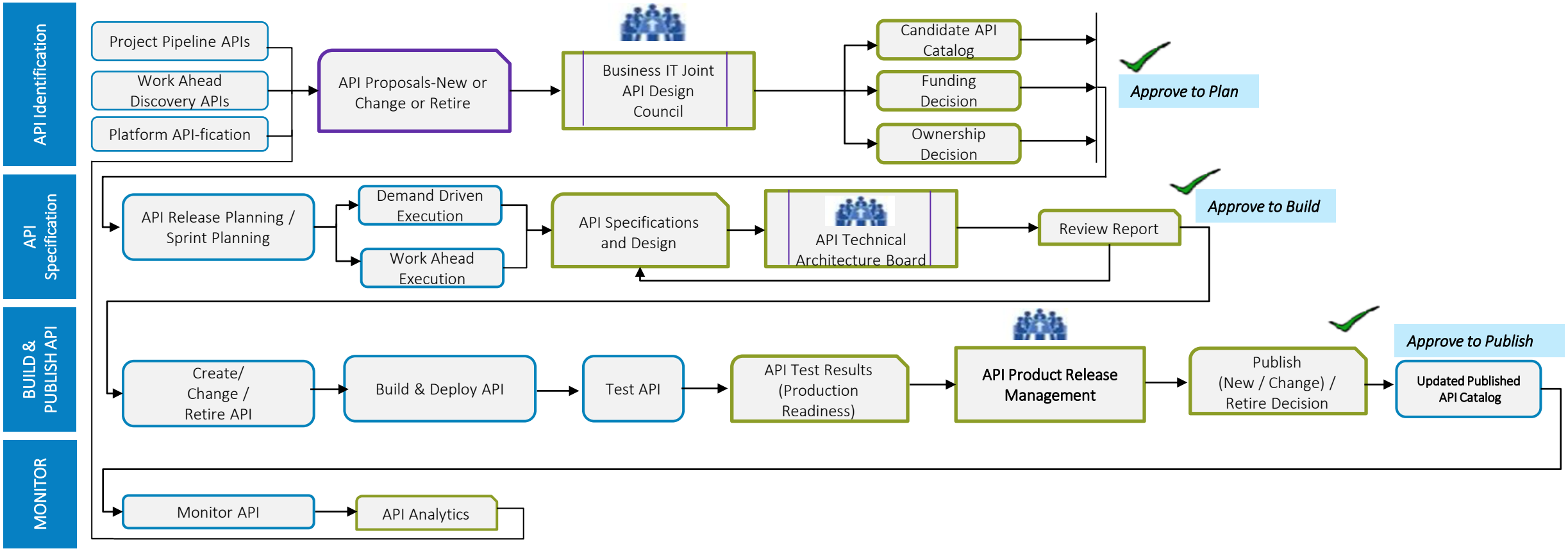
Data Architecture
- Master Data Management
- Reference Data
- Flat Object Structure
- Independent Tables
- Blob Storage

Deployment
- EARs Packaging
- Containerization
- Container Orchestration
- Infra Pipeline – IaC
- Application Pipeline

Re-factoring
- Macro Service to Micro Services
- Serverless
- App Component Dependencies
- Flatten & Refactor Components
- Component Grouping
- API Proxy & Facade

Cross-cutting Concerns
- AuthN & AuthZ
- Service Mesh
- Shared services/domain model
- Log Aggregation & Tracing
- Service Monitoring
- NFRs

Modernization Themes - API First Strategy



Considerations

- Unlocking legacy assets
- API-centric app integration
- Multi-channel user experiences
- B2B integration

- API Developer Portal
- API Discovery
- API Consumption
- Scaling

- API Gateway
- Subscription & Token Auth
- Caching
- CI/CD - DevOps

- Integrations
- Security
- Monitoring
- Analytics

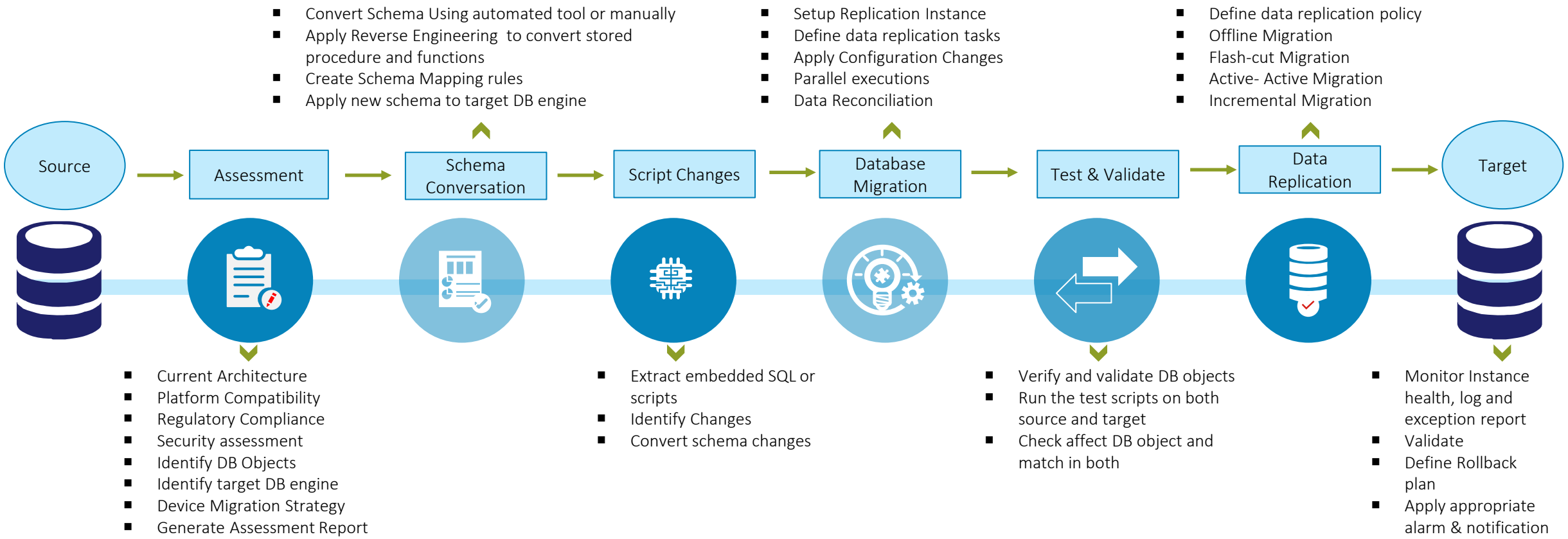
Tools & Services

API Management

Azure Functions

API Gateway

Modernization Themes – DB Modernization



Examples

Homogenous	Heterogenous	Cloud PaaS
Oracle 8x/9x/11x → Oracle 19c	Oracle 12 → MSSQL	Oracle → Azure SQL
MySQL 5.x → MySQL 8	MySQL → Postgres	MS SQL → Azure SQL

Tools & Services


Azure DMS


ADF

Azure SQL

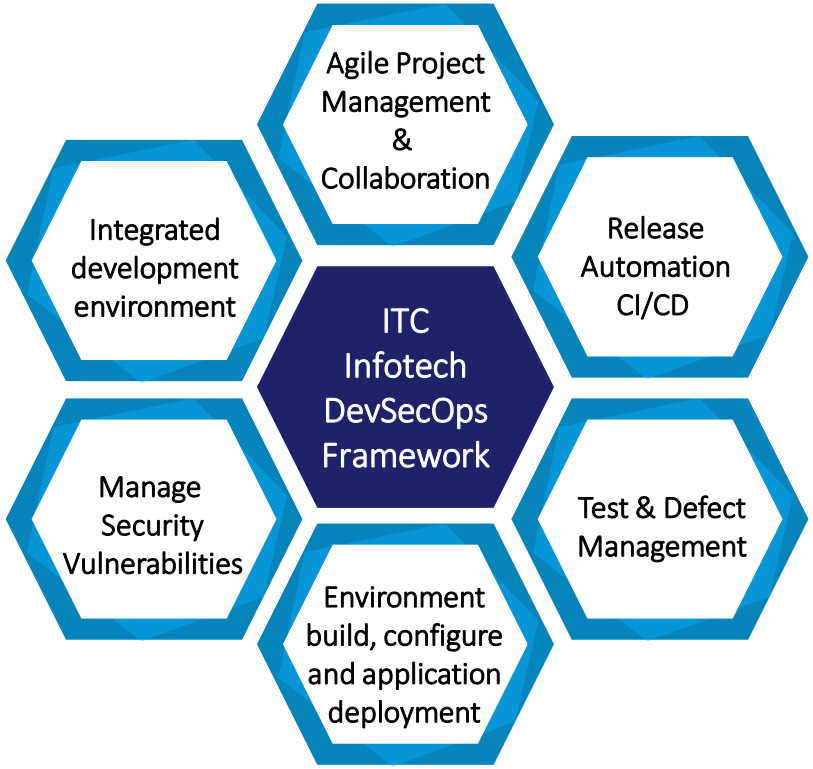
Modernization Themes – DevSecOps

Guiding Principles

- No code DevSecOps - GUI based, automation
- Cloud First - scalable across hybrid Cloud ecosystem
- Secure by Design – RBAC and Policy control
- DevSecOps compliant templates
- Value Stream – AI/ML based analytics

Potential Benefits

- Cloud agnostic DevSecOps platform - GitLab based reusable, scalable Core
- Automating the deployment workflow and change approval process
- Infra-as-code pipeline architecture for reusable library development
- Implementing AI/ML Ops based on need and requirements

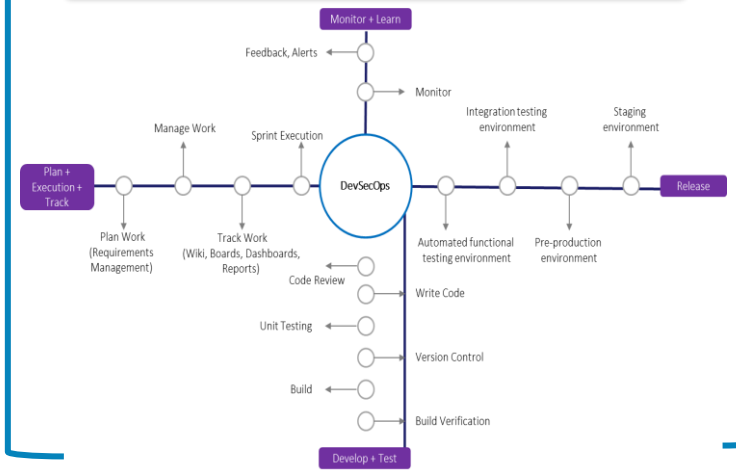


Potential tools that can be leveraged as appropriate

Key Considerations

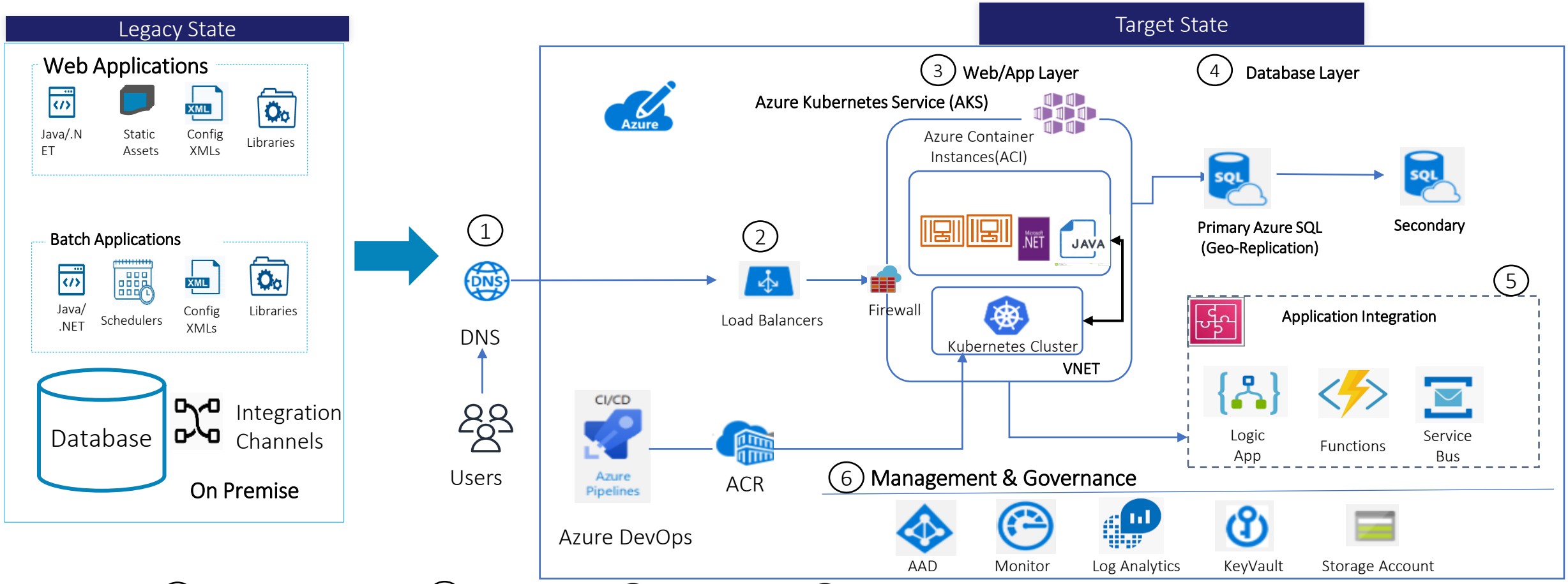
- Customized process as per application type (Custom/COTS)
- Automated provision of infra resource (Iac)
- Use pre-built templatized architecture blueprint library
- Constantly vulnerability scan
- Real time monitoring to track system performance
- Scalable across the application landscape
- Promote reuse
- Rationalization of toolsets
- Future ready for any cloud changes

Course of Actions



Solution Patterns

Solution Pattern - Monolith Java/.NET App to Containerization App



1. Azure DNS will be routing internet traffic to application domain name.

2. Azure Firewall inspects requests coming to CloudFront and perform allow or block as per defined rules. Allowed request passes to Azure Load balancer.

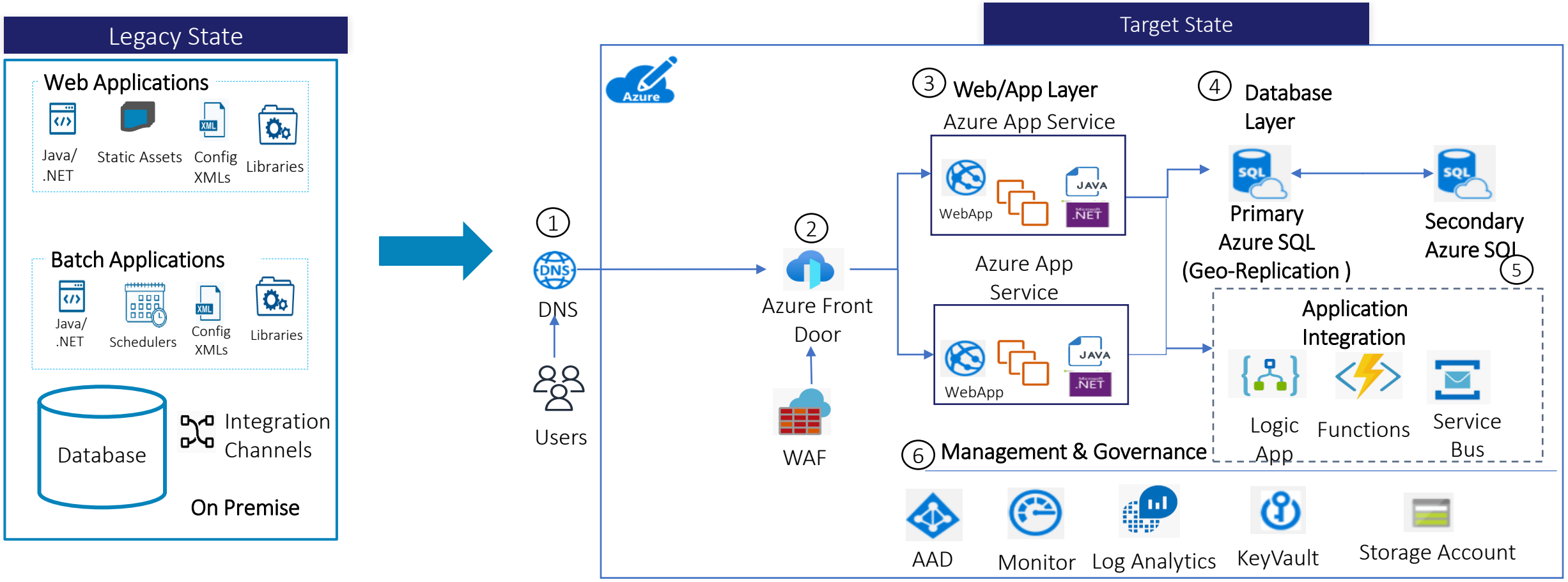
3. The containerized web/app layer will be deployed using AKS service via the ACR

4. Azure SQL is used for application database with Replication option

5. Application Integration can be achieved with multiple Azure offerings. Some of the examples are Logic App , Functions, Event Hub and Service Bus

6. Application resource access managed by IAM roles and policies. Monitor and Log Analytics OMS is used to do logging. Key Vault is used to store Keys and Secrets of Applications. Storage Account is used to store application data to maintain sources of truth and applying life cycle rules.

Solution Pattern - Monolith Java/.NET App to Azure PaaS



1. Azure DNS will be routing internet traffic to application domain name

2. Azure Load Balancer is used in AKS and After creating an AKS cluster, the cluster is ready to use the load balancer

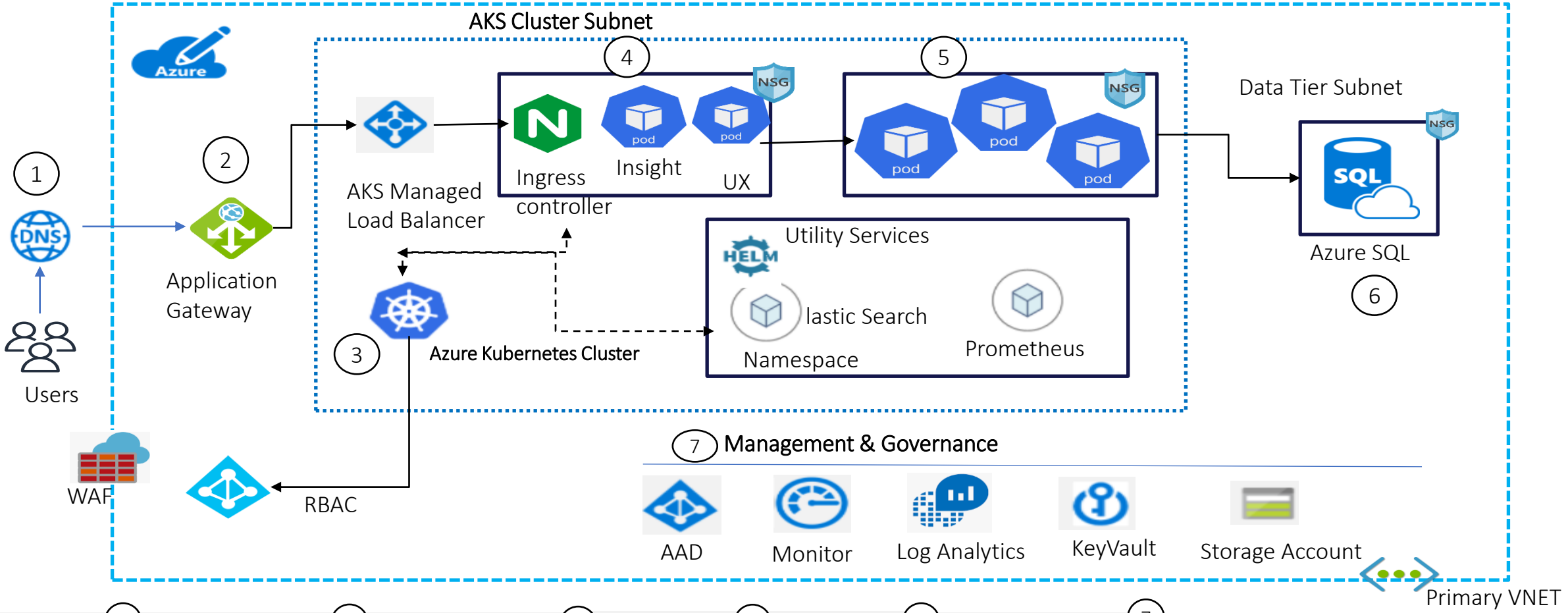
3. The containerized web/app layer will be deployed using AKS with ACI

4. Azure SQL is used for application database with geo-replication

5. Azure Container Registry. Use Container Registry to store private Docker images, which are deployed to the cluster. AKS can authenticate with Container Registry using its Azure AD identity. Note that AKS does not require Azure Container Registry. You can use other container registries, such as Docker Hub.

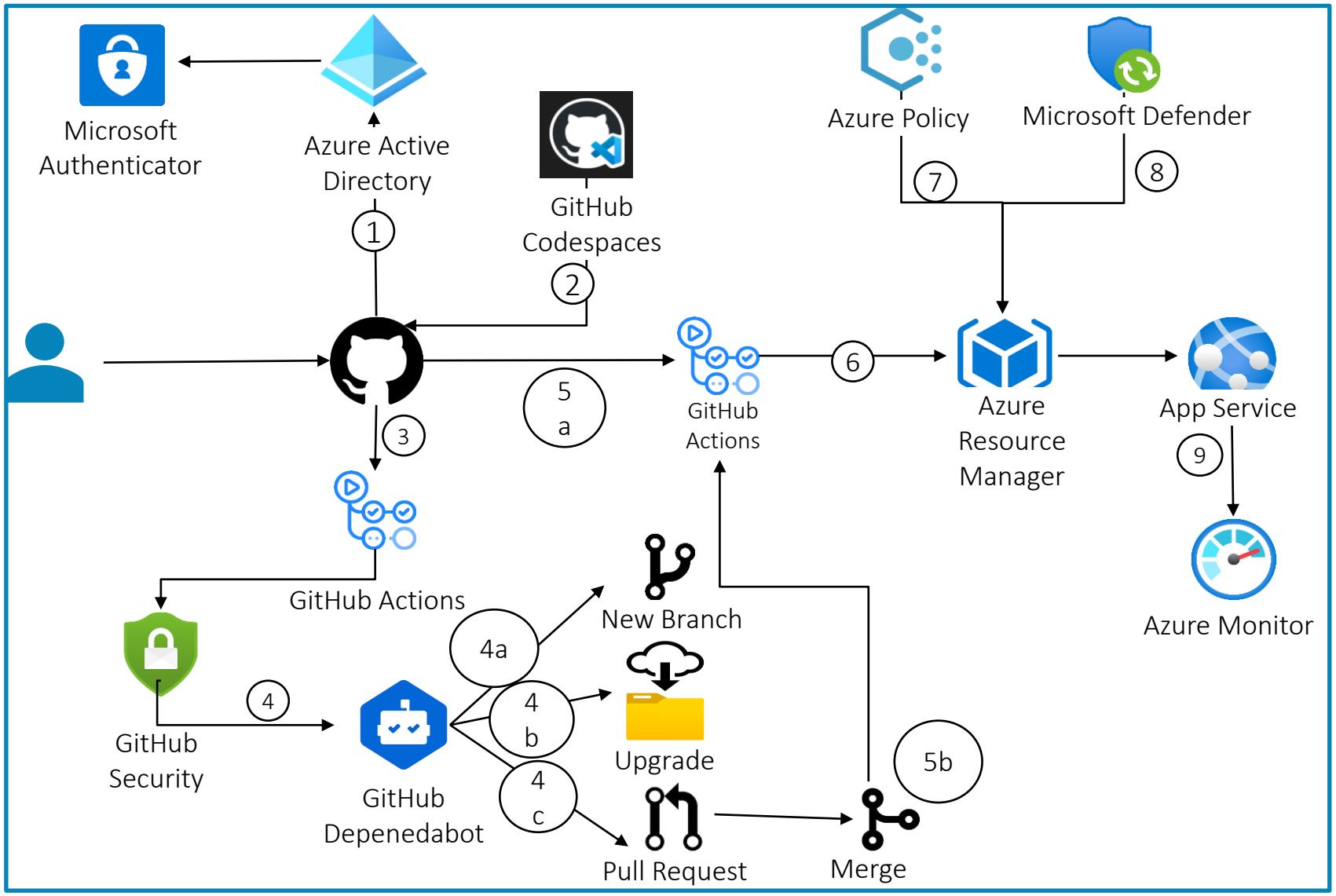
6. Application resource access managed by AAD IAM roles and policies. Azure Monitor and Log Analytics is used to do logging. Key Vault is used to store Keys and Secrets of Applications.
Storage Account is used to store application data to maintain sources of truth and applying life cycle rules.

Solution Pattern - Microservices Architecture on Azure Kubernetes Service



1. Azure DNS will be routing internet traffic to application domain name.
2. API Gateway will centrally manage all request and route to specific resources and block unwanted requests
3. AKS will take the input request and pass on to AKS LB via the Ingress controller
4. AKS will launch the app front end via Ingress Routing Controller
5. Services will run from various AKS PODS
6. Azure SQL Will be used to storage data in the central DB from the AKS nodes apps
7. Application resource access managed by AAD IAM roles and policies. Azure Monitor and Log Analytics is used to do logging. Key Vault is used to store Keys and Secrets of Applications.
Storage Account is used to store application data to maintain sources of truth and applying life cycle rules

Solution Pattern - DevSecOps with GitHub

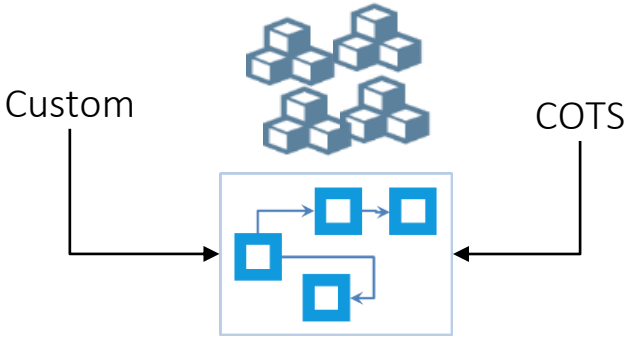


1. GitHub do developer SAML authentication using MS authenticator
2. Developers use pre-built dev env with security extensions
3. On new code commit - GitHub Actions automatically scan the code to quickly find vulnerabilities and coding errors
4. GitHub Dependabot, a DevOps bot agent, automatically detects vulnerability in dependencies
5. Trigger code builds and automated testing through GitHub Actions
6. Deploy build artifacts to Azure App Service while making changes to other cloud resources, such as service endpoints
7. Azure Policy evaluates Azure resources that are in deployment
8. Microsoft Defender for Cloud identifies attacks targeting applications that are running in deployed projects
9. Azure Monitor continuously tracks and evaluates app behavior. When threats materialize, this service sends alerts to start the process of rolling code back to previous commits

Cross cutting Solution Pattern – Handling Technical Debt

Causes for Tech debt

Possible areas to look into it



- Old/EOL Software/Hardware Version
- Poor design/architecture
- Manual testing/Code deploy approach
- Legacy COTS products

Discover

Discover through process and measures

Support Model/Governance/Process

- COTS support programs and cadence calls
- Leverage/engage with enterprise architecture /strategy groups
- Code Linting and Profiling
- Regression and Performance Testing

Tools and accelerators

CAST, sonarqube, SYNOPSYS, JFrog Xray

Remediate/Respond

Execute

Version/software obsolescence

- Rebuild/Migrate to newer technologies
- Evaluate and identify choices for SaaS adoption

Design & Development

- Decoupled architecture and Microservice adoption

Testing & Quality

- Quality Gate and Measures

Deployment

- Container based deployment models

Tools/Standards Adoption

- Adopt Code scanning and product scanning tools

Planning and Delivery

- Product Risk analysis and Mitigation

Cross cutting Solution Pattern – Version Upgrade

Key Considerations

- Downtime and Application Availability
- Reduced Deployment risks
- Slow Rollback (vs) Instant Rollback
- Sticky sessions
- Cost & Operational Overhead
- Observability
- Ability to test live production traffic
- Testing new backend features by using the production load

Best Practices

- Version Lifecyle management
- Backward compatibility checks & scripts - for UI, App layer, DB schema changes
- Continuous integration/continuous deployment/Continuous testing (CI/CD/CT)
- Automated pipelines- Managing Environment
- Operating & Configuration Management Rollback strategy
- Release Automation Tools
- Post Deployment Monitoring
- Infrastructure as Code (IaC)
- Monitoring tools

Patterns

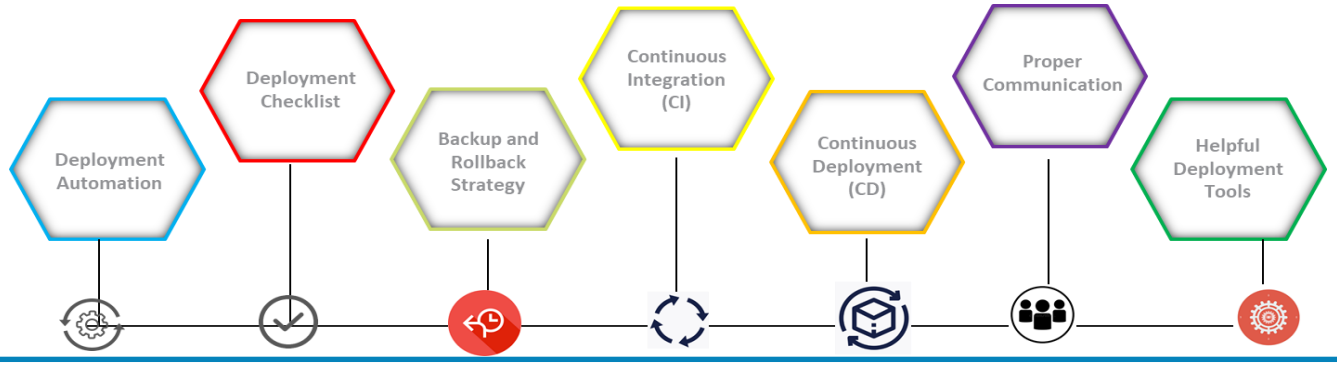
Deployment Pattern

- Recreate
- Rolling Update
- Blue/green
- Canary

Testing Pattern

- A/B
- Shadow

Software Deployment Best practices



TOOLS AND ACCELERATORS

Cloud Native Modernization - Tools & Accelerators

ITC Infotech Tools

Monolith 2 Microservices Assessment tool

Intelligent Monitoring Platform

CSP PaaS offerings

Azure

Azure AD

App Service

Function

Service Mesh

Blob Storage

Service Bus

API management

AKS

ACR

DevOps

DMS

ADF

Azure SQL

Storage Account

KeyVault

Monitor

Log Analytics

3rd Party Tools

Version Control

git

GitLab

GitHub

Secure Code Review

sonarqube

FORTIFY

PMD Code Quality Matters!

CAST Software Intelligence for Digital Leaders

AppScan IBM Security

API Management

Kong

IBM API Connect

DevSecOps

JIRA

Jenkins

Bamboo

JFrog Artifactory

docker

CHECKMARX

IaC and Configuration Management

Packer

ANSIBLE

puppet labs

splunk>

Nagios

dynatrace

APPDYNAMICS

elasticsearch logstash kibana

Web Application Pen Testing

acunetix

HCL AppScan

Infra VAPT and Vulnerability Management

Nessus vulnerability scanner

Qualys

NMAP

Standards

OWASP Open Web Application Security Project

HIPAA

PCI

SANS

CIS

NIST

GOVERNANCE

Migration Governance – Collaboration Model

Assessment	Plan & Design	Migration			Post Migration
Discovery & Assessment →	Wave planning & Design →	Build ⇄	Test →	Deploy →	Hypercare
Review Migration Strategy	Create Wave Plan	Technical Migration	Performance & Security Testing	Data flow cut-over	Resolve Defects Reported in Production
Provide Access to Existing Document(s) & resources	Create Move Groups (MGs)	Application Remediation	Acceptance Testing	Application cut-over	Build Pre-prod using system copy
Check Pre-Requisites for Migration	Product (COTS/3rd Party) Design	Unit Testing	Migration Defect Resolution	Support in Cutover Activities	Document Lessons Learnt
Perform tool and manual discovery	Migration detailed design for MG	Functional Testing	Provide Input for Cutover Planning	Business validation before/after	Knowledge Transition to Support Team
Product (COTS/3rd Party) Specific Migration inputs			Deployment (prod-cutover) Plan Preparation	Migration in PROD	Simulate Disaster Recovery and Failover
Perform assessment and propose disposition				Provide Input for Go/ No-Go Decision	
App Disposition, Multi-Cloud decisions				Technical Go-Live in PROD	

JOINT | ITC | CUSTOMER | VENDOR

ITC INFOTECH

Business-friendly Solutions

THANK YOU

Email: contact.us@itcinfotech.com

Web: www.itcinfotech.com