ITC Infotech Cloud Native Offerings
## ITC Infotech - Service Offerings and Frameworks

<table>
<thead>
<tr>
<th>Strategy &amp; Consulting</th>
<th>Maturity Assessment &amp; Roadmap</th>
<th>Migration Services</th>
</tr>
</thead>
</table>
| - Cloud Native application strategy, planning and implementation approach | - Cloud and Cloud native capabilities assessment  
- Adoption Plan, Reference architecture blueprints | - Application migration on Cloud native capabilities  
- Enhance with Cloud containerization and DevOps |

<table>
<thead>
<tr>
<th>Engineering</th>
<th>Technology</th>
<th>Value Adds &amp; Accelerators</th>
</tr>
</thead>
</table>
| - Application Modernization with cloud native technologies  
- API and Microservices based design and development on cloud  
- Serverless architecture services development | - Azure Cloud native experience  
- Large Scale Implementation experience  
- API Led Digital transformation and Digital Integration Experience | - Implementation Accelerators reduce overall Implementation Effort  
- Microservices Assessment framework  
- Integrated Development & Deployment framework |
Our Cloud Native Solution - Benefits

- Improve Time to Market
- Roll Out New Features Fast
- Eliminate Application Defects
- Improve App Performance
- Automate Application Development
- Easier Application Management
- Reduce App Downtime
- Greater Scalability
ITC Infotech Migration Strategy
Migration Methodology - Phased Approach

Discovery & Assessment
- Discovery
- Assessment

Plan & Design
- Wave Planning
- Design

Migration
- Build
- Test
- Deploy

Post Migration
- Hypercare
- Transition

---

**Tool based Discovery**
- Network, Infrastructure, Server usage etc...

**Assessment Framework**
- Application metadata, interfaces, dependencies etc...

**Manual Discovery**

---

**Wave Planning**

**Design**
- Network Infrastructure
- Application
- Integration

**7R Decision Framework**
- Re-Host
- Re-Platform
- Re-Factor
- Retain
- Retire
- Re-Purchase
- Re-Architect

---

**Migration Factory**
- Build
- Test - Quality Checks
- Deploy
- DevSecOps

**Migration Factory**
- Containerization, Code remediation, refactoring, Native development etc...

---

**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

---

©2022 ITC Infotech. All Rights Reserved.
Our Cloud Native Solution Framework
Cloud Native - Modernization Themes
Modernization Themes – Application Containerization

Evaluate Enterprise Legacy (Monolith) Inventory Applications

Application Containerization

Tools & Services

Gather Application Requirements

- Analyze and gather all the system requirements required for Containerization

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 CI-CD / DevOps Integration for the applications

Feedback & Optimizations
- Review user’s feedback and overall application usability and performance and optimize as and when required.

Performance & Monitoring
- Identity key performance metrics

Test & Rollout to cloud
- Test and Rollout application in Phases

Container Security
- Set up Container Security

©2022 ITC Infotech. All Rights Reserved.
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

**Considerations**

<table>
<thead>
<tr>
<th>Application Architecture</th>
<th>Data Architecture</th>
<th>Deployment</th>
<th>Re-factoring</th>
<th>Cross-cutting Concerns</th>
</tr>
</thead>
<tbody>
<tr>
<td>No Existing Webservices</td>
<td>Master Data Management</td>
<td>EARs Packaging</td>
<td>Macro Service to Micro Services</td>
<td>AuthN &amp; AuthZ</td>
</tr>
<tr>
<td>Existing Macro Services</td>
<td>Reference Data</td>
<td>Containerization</td>
<td>Serverless</td>
<td>Service Mesh</td>
</tr>
<tr>
<td>REST or JMS</td>
<td>Flat Object Structure</td>
<td>Container Orchestration</td>
<td>App Component Dependencies</td>
<td>Shared services/domain model</td>
</tr>
<tr>
<td>SOAP or EJB</td>
<td>Independent Tables</td>
<td>Infra Pipeline – IaC</td>
<td>Flatten &amp; Refactor Components</td>
<td>Log Aggregation &amp; Tracing</td>
</tr>
<tr>
<td>Servlet / JSP</td>
<td>Blob Storage</td>
<td>Application Pipeline</td>
<td>Component Grouping</td>
<td>Service Monitoring</td>
</tr>
<tr>
<td>JSF / Spring Portlet</td>
<td></td>
<td></td>
<td>API Proxy &amp; Facade</td>
<td>NFRs</td>
</tr>
</tbody>
</table>

**Tools & Services**
- Azure WebApp
- Functions
- Service Bus
- Logic Apps
Modernization Themes - API First Strategy

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

Tools & Services
- API Management
- Azure Functions
- API Gateway

©2022 ITC Infotech. All Rights Reserved.
Modernization Themes – DB Modernization

**Assessment**
- Current Architecture
- Platform Compatibility
- Regulatory Compliance
- Security assessment
- Identify DB Objects
- Identify target DB engine
- Device Migration Strategy
- Generate Assessment Report

**Schema Conversation**
- Extract embedded SQL or scripts
- Identify Changes
- Convert schema changes

**Script Changes**
- Setup Replication Instance
- Define data replication tasks
- Apply Configuration Changes
- Parallel executions
- Data Reconciliation

**Database Migration**
- Verify and validate DB objects
- Run the test scripts on both source and target
- Check affect DB object and match in both

**Test & Validate**
- Define data replication policy
- Offline Migration
- Flash-cut Migration
- Active-Active Migration
- Incremental Migration

**Data Replication**
- Monitor Instance health, log and exception report
- Validate
- Define Rollback plan
- Apply appropriate alarm & notification

**Tools & Services**
- Azure DMS
- ADF
- Azure SQL

**Examples**

- **Homogenous**
  - Oracle 8x/9x/11x → Oracle 19c
  - MySQL 5.x → MySQL 8

- **Heterogenous**
  - Oracle 12 → MSSQL
  - MySQL → Postgres

- **Cloud PaaS**
  - Oracle → Azure SQL
  - MS SQL → Azure SQL

©2022 ITC Infotech. All Rights Reserved.
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

ITC Infotech DevSecOps Framework
- Agile Project Management & Collaboration
- ITC Infotech DevSecOps Framework
- Integrated development environment
- Manage Security Vulnerabilities
- Environment build, configure and application deployment
- Test & Defect Management
- Release Automation CI/CD

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

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
- Potential tools that can be leveraged as appropriate

©2022 ITC Infotech. All Rights Reserved.
All trademarks are the registered property of the respective companies and are acknowledged.
Solution Pattern - Monolith Java/.NET App to Containerization App

**Legacy State**

- **Web Applications**
  - Java/.NET
  - Static Assets
  - Config XMLs
  - Libraries

- **Batch Applications**
  - Java/.NET
  - Schedulers
  - Config XMLs
  - Libraries

- **Database**
  - Integration Channels
  - On Premise

**Target State**

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.

©2022 ITC Infotech. All Rights Reserved.
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.
Azure DNS will be routing internet traffic to application domain name.
API Gateway will centrally manage all request and route to specific resources and block unwanted requests.
AKS will take the input request and pass on to AKS LB via the Ingress controller.
AKS will launch the app front end via Ingress Routing Controller.
AKS will launch the app front end via Ingress Routing Controller.
Services will run from various AKS PODS.
Azure SQL will be used to storage data in the central DB from the AKS nodes apps.
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.

Integration Details:

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.
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

  - Custom
  - COTS

  - 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 Lifecycle 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

---

©2022 ITC Infotech. All Rights Reserved.
TOOLS AND ACCELERATORS
Cloud Native Modernization - Tools & Accelerators

ITC Infotech Tools

Monolith 2 Microservices Assessment tool

CSP PaaS offerings

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
Secure Code Review
API Management
DevSecOps
Git
GitLab
GitHub
sonarqube
Fortify
CAST
PMD
Software Intelligence for Digital Leaders
AppScan
Jenkins
Bamboo
docker
CHECKMATE
Packer

JIRA
Nagios
dynatrace
Splunk
Infra VAPT and Vulnerability Management

IaC and Configuration Management

Web Application Pen Testing

OWASP
Open Web Application Security Project
ERM
PCI
SANS
CIS
NIST

All trademarks are the registered property of the respective companies and are acknowledged.
Migration Governance – Collaboration Model

**Assessment**
- Review Migration Strategy
- Provide Access to Existing Document(s) & resources
- Check Pre-Requisites for Migration
- Perform tool and manual discovery
- Product (COTS/3rd Party) Specific Migration inputs
- Perform assessment and propose disposition
- App Disposition, Multi-Cloud decisions

**Plan & Design**
- Create Wave Plan
- Create Move Groups (MGs)
- Product (COTS/3rd Party) Design
- Migration detailed design for MG

**Migration**
- Technical Migration
- Application Remediation
- Unit Testing
- Functional Testing
- Performance & Security Testing
- Acceptance Testing
- Migration Defect Resolution
- Provide Input for Cutover Planning
- Deployment (prod-cutover) Plan Preparation

**Post Migration**
- Data flow cut-over
- Application cut-over
- Support in Cutover Activities
- Business validation before/after
- Migration in PROD
- Provide Input for Go/No-Go Decision
- Technical Go-Live in PROD
- Resolve Defects Reported in Production
- Build Pre-prod using system copy
- Document Lessons Learnt
- Knowledge Transition to Support Team
- Simulate Disaster Recovery and Failover

**ITC CUSTOMER VENDOR**