Agile DevOps Cloud Assessment Template

Assessments - Looking to see if they can deliver faster with higher quality

- Development Practices (plan on a faster pace) to follow an iterative model
- Embed code quality. Security inside of practices to enable shift-left
- People / Process / Tools
- Address 3 or 4 areas for a particular focus group and get to the final answer
- What is the business value (business case) for the assessment

Assessment Practice -

Portfolio Management - overall strategy, envision and product roadmap with business goals, master backlog

- How do we validate the ideation and process of vetting that idea
- How do we prioritize the backlog (define the weight) of a list of ideas to determine
- Application Process Management - to get it done thru Engineering
- Capacity management - how do business measure the capacity required for a deliverable vs. cost
- Process for defining a portfolio item, and how is its lifecycle managed
- How do we measure success (planned vs. delivered) and revenue expected vs. revenue actual
- Epics - Themes to provide query script to analyze
- *** need to come up with a plan to define scorecard and provide standard scoring model

Program Management - understanding appropriate requirements divided into phases of prod development, release management

- Provide high level engagement with business team on what is agreed upon and what to be delivered over the course of a timeline
- Sequence items and manage change control process for change of business priority vs. technology strategy
- Define standards and enforce models for scrum teams
- Own end-2-end delivery and lifecycle management of the product (including operationalization and monitoring)
- Features / FeatureSets (lifecycle management, plan release vs. actual release, change control items (internal / external), release planning)
- Capacity management - what is the billable ratio dev and friends (time allocation, ratio values, measurement)
- Delivery model supported (How are the scrum teams defined, organized, environment strategy, metrics / measurement, release overview, production oversight)

Scrum / XP / Kanban - full stack engineering team responsible for end-2-end iterative delivery based on incremental value addition

- When do they use what (Scrum, XP, Kanban)
- Structure of team model to support (# of resources, # area of focus, accountability factors, full stack vs. shared services)
- Sprint ceremonies (process definition, delivery frequency, outliers, metrics, change control management, user story definition, sizing)
- Lifecycle management, issue type relationships (Epic, Feature, Story, Bug, Task), integration points w/tools
- Sprint model, integration with release or HIP
- Definition and Validation of a story / task based on agreed iteration (test delivery model)
- Effort estimations / user stories - (standardize)

Architecture / Developer / Test Engineering - Focus on engagement model and roles / responsibilities across the delivery model

- Org summary / overview and responsibilities (Architecture, Dev, Test) - metrics, governance, 3rd party apps / components, documentation
- Design Patterns - (12 factor app), Developer / Test Model (BDD,/TDD)
- Developer Onboarding - tech stack overview, application review, engagement model, high level team structure and org structures, role base access
- Developer Productivity (assignment of work, environment setup, workspace management, repo management, branching model, build definitions / package / publish / deployment logic, artifact versioning, artifact management, security practices, unit test, code quality, measure dev efficiency, lead time / overhead, API contract docs)
- Non Functional Requirement (performance, security, infrastructure, platform) - activities and consideration to implement the solution
- Test Productivity (documentation, ALM, test code artifacts, test framework, test automation, automation strategy, non functional testing strategy, requirement traceability Matrix, test data lifecycle management, production access review, manual vs. automated coverage)
- Secure Product Development Lifecycle - (security assessment, security architecture, penetration testing, data encryption, logging / masking, secure threat modeling, security patch management, secure threat lifecycle management, vendor and 3rd party app validation)

Tool Chain / Release Management

- Release Definition (roles / responsibilities, event types - app/db/infra/security/middleware, engagement model, lifecycle management, change control board, parallel release trains, gate checks, oversight model, business control plan)
- Release Delivery Framework (governance, versioning, planning, registration, merge, lock, certification activities, certification triage, Go/No Go, deploy strategy (canary, rolling, active/passive, blue/green), backfill environments, operations hand-over, roll back strategy)
- Environment Strategy (release event mapping, RBAC, life cycle management, provisioning, environment types, # of VMs / Containers Cloud, vulnerability management, availability metrics, network topology)
- Tool Chain Overview (tech stack, versions, architecture, configurability, scalability, integration points, plugins / custom adapters, continuous workflow model, traceability, instrumentation, audit)
    - Requirement Management (app type, app version, architecture, workflow scheme, screen scheme, notification scheme, permission scheme, issue type layering, metrics)
    - Source Control Management (project mapping, repo structures, repo permissions, on-boarding, branching model, branching lifecycle, artifact version lifecycle management, database versioning, app/env config management, gates, notification model, code review mechanism, merge types, roll backs, integration points)
    - Build (build tool, build version, build architecture, customization, build master/slave configuration, build dependency management, build optimization, build scheduler, build types, integration points, static analysis, code coverage, notification model)
    - Package (package tool, package version, packaging architecture, access control management, packaging types, package versioning, package artifact lifecycle, package dependency, integration points)
    - Publish (publish tool, publish version, package life cycle management, publish versioning)
    - Deploy (deploy tool, deploy version, deploy architecture, deploy strategy, credential management, tokenization, deploy activation, deploy validation, deploy downtime, notification, integration points)
    - Security (security governance, static application security testing, dynamic application security testing, artifact scans, 3rd party compliance / validation, 3rd party artifact scans, tool architecture, integration points, gates, notification)
    - Test (automation / performance / monitoring tools and code coverage, test management tools, integration points, validation w/function/integration/regression/performance, external lab, failure injection strategy, validate internal / health check service, root cause analyzer)
- Tool Chain Continuous Integration / Delivery (pipeline architecture, pipeline scalability, configurability, tech stack, CI delivery model mapping (invocation, metrics/dashboard, gates), CD delivery model (invocation, metrics/dashboard, gates)
    - Tool chain architecture (non-prod vs. prod tool matrix), design, platform and health checks
    - Process Lifecycle mapping of planning, dev, build, deploy and test activities
    - Alignment with Configuration Management and Infrastructure as Code platforms (VMs, Containers, FaaS)
    - Production roll out strategy alignment (tool version, accelerators, deploy strategy mapping)
    - Production Issue Management

Infrastructure / Operations/ Cloud

- Technology Stack, Data Center/ Environment Strategy, Service Catalog, Foundational Services, Self Service Capabilities, Lifecycle management, High level visual architecture diagram/Topology Diagrams/Network (ICCR) , Prod / Non Prod Ratio and configuration, business continuity(CMDB), Cost of ownership, Vendor Management/License Management, Ongoing maintenance/support
    - Infrastructure as Code, Infrastructure as a Service, Platform as a Service, Container as a Service, Functional as a Service,
    - SRE - Incident Management/Logs Monitoring/Salability/Performance Management/Partner Technical Services/Self healing,Perimeter Check (health, infra, environment) for resiliency, RTO, RPO
    - Security policies and procedures (RBAC/IAM/Vulnerability Scans/Authorization/Authentication/Image Scans
    - Regulatory Compliance/Back up/ Retention policies/Replication/
    - Adoption Model/Hybrid architecture (SSO) common identity.
- Cloud Types / Selection models (AWS, Azure, Salesforce, SAP 4/Hana), Services (IAM, Roles, Server Configuration, Virtual Networks, Auto Scaling, Subnets, Monitoring, Load Balancing, Storage, Deploy Tools, Security, Caching, CDN, Repository Management, CI/CD Pipeline, API auditing, API Gateway, DNS, Virtual Private Network, Resource Management, Trusted Advisor, Database Strategy, Templates), Deployment Model(rolling, blue/green, canary),Cloud migration(lift & Shift, Refactor, Retire),Tagging strategy, Cost optimization, Fault tolerance architecture design