

Remote Cloud Engineer Assessment Test

Candidate's Name: Reena Smile Janga

GitHub Repository (Technical Exercise):

<https://github.com/ReenaSmileJanga26/bluebird-cloud-exercise>

PART ONE

1. Are you currently employed?

Yes, I am currently employed. I am exploring this opportunity because it aligns strongly with cloud infrastructure ownership, security-minded design, and DevSecOps (Development, Security, and Operations) delivery.

2. How many hours can you devote weekly to the Company?

I can devote 40 hours per week, with overlap across time zones as needed. I am comfortable supporting critical production events outside standard hours with proper planning and rotation.

3. Why do you think you fit the role well? Please provide examples of your skills and experiences that align with the requirements of this role.

I fit this role well because I combine strong Infrastructure as Code (IaC) execution with secure cloud architecture and operational discipline. For example: (1) I design multi-tier networks using Amazon Virtual Private Cloud (Amazon VPC), public/private subnets across Availability Zones (AZs), and least-privilege security groups. (2) I build modular Terraform code (network, security, compute, database, storage, monitoring) with parameterization and reproducible environments. (3) I implement Continuous Integration and Continuous Delivery (CI/CD) pipelines with linting, security scanning, cost visibility, and gated production deployment. (4) I focus on observability using Amazon CloudWatch metrics, logs, and alarms, and I document runbooks and trade-offs for maintainable operations.

4. Have you worked from home before?

Yes. I am comfortable working remotely with clear written communication, proactive status updates, and disciplined delivery across distributed teams.

5. What interests you about this role?

This role is attractive because it combines hands-on cloud engineering with security and reliability. I enjoy building scalable, automated platforms and making them easy to operate through good observability, automation, and documentation.

6. What motivates you?

I am motivated by ownership and measurable impact: shipping reliable systems, reducing risk through automation, improving security posture, and enabling teams to move faster with confidence.

7. Are you seeking employment in a company of a specific size, such as a small startup, medium-sized Company, or large corporation? Please explain your preference.

I am open to different company sizes, but I tend to perform best in small-to-medium environments where engineering has direct ownership and fast feedback loops. I like roles where I can contribute end-to-end (design, build, automate, operate) and where outcomes are visible.

8. Describe your experience working with a culturally diverse group of people

I have worked with culturally diverse teams across locations and time zones. I keep communication structured (clear written updates, shared docs, action items), confirm assumptions early, and align on working agreements (response times, escalation paths, meeting norms) to avoid misunderstandings.

9. (a) Describe your experience designing and implementing solutions across multiple cloud providers (AWS, Azure, GCP). What factors influence your decision to choose one provider over another for specific workloads? (b) Walk us through how you would architect a highly available, fault-tolerant application that spans multiple availability zones or regions. What are the key considerations and trade-offs?

(a) I have primary hands-on experience in Amazon Web Services (AWS) and working knowledge of Microsoft Azure and Google Cloud Platform (GCP) concepts. Provider choice depends on existing ecosystem and skills, compliance requirements, managed service maturity, cost model, networking constraints, and regional presence/latency. (b) For high availability across AZs, I use an internet-facing Application Load Balancer (ALB) in public subnets, run stateless compute (for example, Amazon Elastic Container Service on AWS Fargate) across private subnets in at least two AZs, and use a managed database with Multi-AZ failover. Health checks and auto scaling are configured at the service level. For multi-region, I add Domain Name System (DNS) failover (for example, Amazon Route 53), replicate data (read replicas or async replication depending on database), and define Recovery Time Objective (RTO) and Recovery Point Objective (RPO). Trade-offs include cost, operational complexity, and data consistency.

10. (a) Describe a complex cloud migration project you've led or participated in. What migration strategy did you use (rehost, replatform, refactor) and why? What challenges did you encounter? (b) Compare and contrast Terraform, CloudFormation, and ARM templates. Which do you prefer and why? Provide an example of a complex infrastructure component you've built using IaC.

(a) In a complex migration, I typically start with application discovery, dependency mapping, and landing-zone readiness (networking, identity, logging). A common approach is replatforming: moving workloads to managed services (for example, containers and managed databases) while keeping the application architecture mostly intact, because it reduces operational overhead without the time/cost of a full refactor. Challenges usually include network connectivity, identity/permissions, data migration cutovers, and aligning rollback plans with business downtime constraints. (b) Terraform is cloud-agnostic, has a strong module ecosystem, and supports consistent patterns across providers. AWS CloudFormation and Azure Resource Manager (ARM) templates are provider-native and integrate deeply with their ecosystems. I prefer Terraform when portability and modular reuse matter. Example: building a complete multi-tier stack with VPC, ALB, ECS Fargate service, Multi-AZ Amazon Relational Database Service (Amazon RDS), Amazon Simple Storage Service (Amazon S3), least-privilege Identity and Access Management (IAM), and CloudWatch alarms, all as reusable

modules.

PART TWO

1. (a) Design a complete CI/CD pipeline for a microservices application. What tools would you use, and how would you handle deployment across multiple environments (dev, staging, production)? (b) How do you approach configuration management and secrets management in cloud environments? What tools and best practices do you follow?

(a) I use GitHub Actions for CI, container builds with Docker, a registry such as Amazon Elastic Container Registry (Amazon ECR), and deploy to Amazon Elastic Container Service (ECS) or Kubernetes. The pipeline includes: formatting/linting, unit tests, static analysis (SAST - Static Application Security Testing), dependency scanning, container image scanning, and then environment promotion. Dev deploys automatically on merge; staging deploys on release candidate; production deploys require manual approval via GitHub Environments and change review. (b) For configuration, I keep non-sensitive config in versioned files or environment variables, and store secrets in AWS Secrets Manager or AWS Systems Manager Parameter Store (SecureString). I avoid plaintext secrets in code, rotate credentials where possible, scope IAM permissions to least privilege, and prefer workload identities (OIDC - OpenID Connect) rather than long-lived keys in CI.

2. Share a challenging technical issue you've encountered and how you resolved it, focusing on your analytical and critical thinking skills.

A common challenging scenario is intermittent HTTP 502/504 errors in front of a load balancer. I approach it by forming hypotheses and validating quickly: check load balancer metrics (target errors, latency), verify health check behavior, inspect application logs and container restarts, and confirm network policies (security group rules and routes). In one case, the root cause was an overly aggressive health check timeout combined with slow startup. Fixes included tuning health check thresholds/timeouts, adding readiness checks in the app, and adjusting auto scaling cooldowns. I then documented the incident with a clear root cause analysis and preventive actions.

3. How do you handle feedback and criticism from supervisors or colleagues?

I treat feedback as input to improve outcomes. I clarify the expectation, ask for concrete examples, and propose an action plan with measurable follow-ups. If I disagree, I explain my reasoning with data and stay focused on the shared goal.

4. What is your greatest strength? How will it help you in this role?

My greatest strength is structured problem-solving with strong ownership. In cloud engineering, that means I can design a secure architecture, automate it end-to-end, and then operate it reliably with monitoring, documentation, and continuous improvement.

5. We want to understand your long-term career goals and how this position aligns with them. Please share your aspirations and how you see this role contributing to your career path.

My long-term goal is to grow into a senior cloud/platform engineering role where I own scalable, secure infrastructure and DevSecOps standards. This position aligns well because it involves designing systems, automating deployments, improving reliability, and contributing to security and cost governance.

6. What do you know about our Company?

From the information available in the role and public materials, my understanding is that your company values dependable delivery, security, and customer outcomes. For a real engagement, I would review your product offerings, target customers, security/compliance requirements, and current cloud stack so I can align early with the business priorities.

7. How do you see yourself contributing to our Company's continued success and growth in this role?

I can contribute by building repeatable IaC modules, secure-by-default networking patterns, and reliable CI/CD workflows that reduce deployment risk. I also focus on FinOps (Cloud Financial Management) practices such as cost visibility in pull requests and regular drift/cost reviews so the platform scales sustainably.

8. In your opinion, what are the biggest challenges facing our Company in the market?

Common challenges include maintaining security and compliance while moving fast, controlling cloud costs as usage grows, and building reliable systems that can handle traffic spikes without over-provisioning. The solution is a mix of automation, observability, and disciplined change management.

9. Discuss your familiarity with recent technological trends and how they can be leveraged within the industry.

Key trends include containerization and managed container platforms, GitOps (Git-based operations) workflows, policy-as-code for security guardrails, supply-chain security (SBOM - Software Bill of Materials and artifact signing), and stronger observability (logs, metrics, traces). Leveraging these trends improves deployment safety, auditability, and operational efficiency.

10. Your application experiences a 10x traffic spike. Walk us through your immediate response and the long-term architectural changes you plan to implement to handle such scenarios.

Immediate response: confirm impact via dashboards, scale out stateless services (increase ECS task count), verify ALB health and error rates, and protect the database by enabling connection pooling and limiting expensive queries. If needed, temporarily raise service limits and enable rate limiting.

Long-term: add caching (for example, Amazon ElastiCache), use a Content Delivery Network (CDN - such as Amazon CloudFront) for static content, decouple heavy work using queues, introduce load testing, and set auto scaling based on CPU and request metrics. I would also review database indexing, read replicas, and implement Web Application Firewall (WAF) rules for resilience.

PART THREE

1. What tools or strategies are most effective for communication and collaboration in a remote work environment?

Clear written communication is the foundation: shared documents, well-maintained tickets, and concise status updates. Tools typically include Jira for work tracking, Slack/Teams for communication, and Zoom/Meet for calls. Strategies include asynchronous updates, defined escalation paths, documented runbooks, and regular retrospectives.

2. If you were hiring for this role, which key personality traits and qualities would you look for?

I would look for ownership, strong communication, calm decision-making under pressure, curiosity, and a security-first mindset. I would also value the ability to document decisions and build maintainable automation.

3. How do you approach problem-solving as a team member versus an individual contributor?

As an individual contributor, I move quickly with clear hypotheses and documented findings. As a team member, I align on goals and constraints first, share options with trade-offs, and ensure decisions are recorded so everyone can execute consistently.

4. How do you prioritize and manage your workload in a fast-paced environment?

I prioritize by impact and urgency: incidents and customer-impacting issues first, then delivery deadlines. I break work into small milestones, communicate risks early, and keep a visible backlog so stakeholders can see progress and trade-offs.

5. What specific skills or experiences do you hope to gain from this role to help you achieve your career goals?

I want to deepen my experience operating secure, scalable systems end-to-end, including advanced security controls, performance optimization, and production-grade incident response. I also want to strengthen FinOps practices for cost-aware scaling.

6. Imagine that you told a client you would be there at 10 am. It is now 10:30 am, and you won't finish your job until 11:30. You have a lunch meeting with another client at noon, followed by another job at 1:15 pm. How would you handle this situation?

I would communicate immediately: notify the 10 am client of the delay, provide a realistic updated completion time, and set expectations. I would also proactively message the noon client to adjust the lunch meeting if needed or reschedule. If possible, I would delegate or split work to meet commitments. The key is early communication and resetting timelines before the impact compounds.

7. In your experience, how frequently does the following problem occur: employees being afraid to express disagreement with their managers?

It occurs occasionally in many organizations, especially where psychological safety is low. Healthy teams encourage respectful disagreement and evidence-based decisions. I try to create a safe environment by inviting input, asking for risks/concerns, and focusing on outcomes rather than blame.

8. Your experience working on projects involving a consortium of companies is valuable. Please briefly share your experience in this area.

In consortium-style projects, success depends on governance and clear interfaces. I have experience coordinating across multiple stakeholders, aligning on requirements and timelines, documenting decisions, and managing integrations through well-defined contracts and change control.

9. How much would you request per hour?

I would request USD 30 per hour as a starting point, and I am open to discussion based on scope, expectations, and long-term engagement.

10. When would you be available to start if hired?

I can start in approximately two weeks, depending on notice period and onboarding requirements. I can also begin part-time onboarding sooner if needed.

TECHNICAL EXERCISE SUBMISSION

Technical Exercise Repository

<https://github.com/ReenaSmileJanga26/bluebird-cloud-exercise>

Summary of the implemented solution

The repository includes a resilient multi-tier web application infrastructure implemented on AWS using Terraform, with a focus on security, availability, scalability, and DevSecOps automation. The design uses an internet-facing Application Load Balancer in public subnets, an ECS Fargate service in private subnets across two Availability Zones, and an Amazon RDS PostgreSQL database deployed in private DB subnets with Multi-AZ enabled. Static assets are stored in an encrypted Amazon S3 bucket with lifecycle rules. Secrets are stored in AWS Secrets Manager, and logs/metrics are captured via Amazon CloudWatch. The CI workflows include Terraform formatting/validation, linting and security scanning, cost estimation using Infracost, and scheduled drift detection.

Included deliverables (as per exercise)

IaC code organized in modules, variables and outputs, deployment scripts, GitHub Actions pipelines, architecture diagram(s), README with deployment/testing/cleanup steps and cost estimate, and supporting documents (DECISIONS.md, IMPROVEMENTS.md, SCENARIOS.md).