Manual QA Engineer Take-Home Assignment

Part 1: Test Case Design

ID	Description	Pre-conditions	Steps	Expected Result	Priority	Category
1	Verify VPC can be created with valid inputs	User is logged in, account has <5 VPCs in region	1. Navigate to VPC creation form 2. Enter valid VPC name 3. Enter valid CIDR block 4. Select region 5. Enable DNS resolution 6. Click "Create VPC"	VPC created successfully within 30 seconds, success message displayed	High	Functional
2	Verify VPC can be created with 1-character name	User is logged in	1. Navigate to VPC creation form 2. Enter VPC name "a" 3. Enter valid CIDR block 4. Select region 5. Click "Create VPC"	VPC created successfully	Medium	Functional
3	Verify VPC can be created with /16 CIDR block	User is logged in	1. Navigate to VPC creation form 2. Enter valid VPC name 3. Enter CIDR block "10.0.0.0/16" 4. Select region 5. Click "Create VPC"	VPC created successfully	High	Functional
4	Verify VPC can be created with /28 CIDR block	User is logged in	1. Navigate to VPC creation form 2. Enter valid VPC name 3. Enter CIDR block "10.0.0.0/28" 4. Select region 5. Click "Create VPC"	VPC created successfully	High	Functional
5	Verify system rejects	VPC with CIDR "10.0.0.0/16"	1. Navigate to VPC creation form 2. Enter valid VPC	Error message "CIDR block	High	Functional

	overlapping CIDR blocks	already exists in region	name 3. Enter CIDR block "10.0.1.0/24" 4. Select same region 5. Click "Create VPC"	overlaps with existing VPC"		
6	Verify system rejects invalid CIDR formats	User is logged in	1. Navigate to VPC creation form 2. Enter valid VPC name 3. Enter invalid CIDR "10.0.0.0" 4. Select region 5. Click "Create VPC"	Real-time validation error "Invalid CIDR format"	High	UI/Functional
7	Verify system handles concurrent creation requests	User has 4 existing VPCs in region	1. Open two browser sessions 2. In both sessions, attempt to create VPC with valid parameters 3. Submit both requests simultaneously	One VPC created successfully, other fails with "VPC limit reached" message	Medium	Performance
8	Verify VPC can be created with DNS resolution and hostnames disabled	User is logged in	1. Navigate to VPC creation form 2. Enter valid VPC name and CIDR 3. Ensure both DNS options are unchecked 4. Click "Create VPC"	VPC created successfully, DNS resolution disabled	Medium	Functional
9	Verify VPC creation succeeds with a 255-character name	User is logged in, account has <5 VPCs in the region	1. Navigate to VPC creation form 2. Enter a 255-character name (e.g., "a" * 255) 3. Enter valid CIDR (10.0.0.0/16) 4. Select a region 5. Click Create VPC	VPC is created successfully	Medium	Functional
10	Verify system rejects names with invalid special characters	User is logged in	1. Navigate to VPC creation form 2. Enter name "test@vpc" (invalid character @) 3. Enter valid CIDR (192.168.0.0/24) 4. Select a region 5. Click Create VPC	Real-time validation error: "Name can only contain alphanumeric, hyphens, and underscores."	High	UI/Functional

11	Verify system rejects non-private IP ranges (e.g., public IPs)	User is logged in	1. Navigate to VPC creation form 2. Enter valid name 3. Enter CIDR 8.8.8.0/24 (public Google DNS range) 4. Select a region 5. Click Create VPC	Error: "CIDR must be in private IP ranges (10.0.0.0/8, 172.16.0.0/12, 192.168.0.0/16)."	High	Security
12	Verify system allows duplicate VPC names (if not enforced)	A VPC named "prod-vpc" already exists	1. Navigate to VPC creation form 2. Enter name "prod-vpc" 3. Enter valid CIDR (172.16.0.0/16) 4. Select a region 5. Click Create VPC	If duplicates allowed → Success. If unique names enforced → Error: "VPC name already exists."	Medium	Functional
13	Verify system rejects submission if required fields are blank	User is logged in	1. Navigate to VPC creation form 2. Leave Name and CIDR fields empty 3. Click Create VPC	Error: "Name and CIDR are required fields."	High	UI/Functional
14	Verify VPC creation time stays under 30 sec during peak traffic	Simulate high load (e.g., 50 concurrent requests)	1. Use a load-testing tool to send 50 concurrent VPC creation requests 2. Measure response time for each request	All requests complete in <30 sec	Medium	Performance
15	Verify system enforces 5-VPC limit per region	Account already has 5 VPCs in us-east-1	1. Navigate to VPC creation form 2. Enter valid name and CIDR 3. Select us-east-1 4. Click Create VPC	Error: "Maximum of 5 VPCs per region reached."	High	Functional

Part 2: Bug Investigation and Reporting

Issue 1: "VPC Creation Sometimes Fails"

Bug Report

- Title: Intermittent VPC creation failure with CIDR 10.0.0.0/16
- Description: VPC creation fails randomly with error "Creation failed - please try again"
- Steps to Reproduce:
    1. Navigate to VPC creation form in us-east-1
    2. Enter valid VPC name
    3. Enter CIDR block "10.0.0.0/16"
    4. Click "Create VPC"
    5. Repeat multiple times during peak hours
- Expected: Consistent successful creation
- Actual: Intermittent failures
- Environment: Staging, us-east-1
- Severity: High
- Priority: High

Investigation Analysis

seeing random failures when users try to create VPCs, especially during busy periods. My hunch is the system is either getting overloaded when multiple people try to claim IP ranges at the same time. The problem seems centered around the VPC creation service . we should keep an eye on server performance during peak times, scan the database, and test whether smaller network blocks work more reliably than larger ones.

These random failures create real headaches for users who suddenly lose their work. So we'll eventually need to rework how the system handles IP allocations to prevent this for good.

Issue 2: "DNS Settings Not Working"

Bug Report

- Title: DNS resolution not functioning despite being enabled
- Description: Instances in VPC cannot resolve domain names when DNS resolution is enabled
- Steps to Reproduce:
    1. Create VPC with DNS resolution enabled
    2. Launch instance in VPC
    3. Attempt DNS query from instance
- Expected: DNS queries succeed
- Actual: DNS queries fail
- Environment: Staging, us-west-2
- Severity: High
- Priority: High

Investigation Analysis

> The DNS failures likely come from misconfigured Route53 endpoints, blocked traffic in security groups. This affects DNS services and network setup. To verify, check DNS settings, route tables, and test different instances. Since apps depend on DNS, this is a critical issue.

Issue 3: "CIDR Validation Inconsistency"

Bug Report

- Title: CIDR validation inconsistency between UI and backend
- Description: UI accepts invalid CIDR (10.0.0.0/15) but creation fails
- Steps to Reproduce:
    1. Enter CIDR "10.0.0.0/15" in UI
    2. Observe no validation error
    3. Submit form
- Expected: Consistent validation at UI and backend
- Actual: UI accepts, backend rejects
- Environment: Staging and production
- Severity: Medium
- Priority: Medium

Investigation Analysis

> The UI and backend disagree on correct CIDR formats - the UI misses some checks that the API does. This lets users enter invalid ranges that later get rejected, creating frustration when submissions fail. While the system still works the backend blocks bad inputs, the poor experience needs fixing rules across both layers to give consistency. Testing shows this happens with certain CIDR sizes that pass UI checks but fail API validation.

Part 3: End-to-End Testing Strategy

Test Strategy Document

- **Testing Approach:**

  We'll validate multi-tier app deployment using VPC, focusing on network isolation, security, and connectivity with both manual and automated testing.
- **Objectives:**

  Verify VPC, network isolation, security groups, component connectivity, internet/NAT access, and load balancer configuration.
- **Key Scenarios:**

  Test VPC/subnet creation, gateway functions, security rules, load balancer setup, database access, and full app functionality.
- **Risks:**

  High: Network misconfigurations exposing data. Medium: Performance issues. Low: UI problems.
- **Success Metrics:**

  All components deploy correctly with proper isolation, security, and performance.

Test Plan

- **VPC Setup:**

  Create VPC (10.0.0.0/16) and verify CIDR in console.
- **Subnets:**

  Create public (10.0.1.0/24), private (10.0.2.0/24), and database (10.0.3.0/24) subnets and validate their AZs/CIDRs.

- **Gateways:**

  Attach Internet Gateway and verify public route table. Deploy NAT Gateway and check private route table.
- **Security:**

  Configure security groups and validate traffic rules between tiers.
- **Load Balancer:**

  Deploy ALB in public subnet and test health checks/access.
- **Application:**

  Launch 2 app instances in private subnet and verify ALB/DB connectivity.
- **Database:**

  Set up RDS in database subnet and test app connections.
- **Rollback:**

  Document teardown steps and verify complete cleanup.

Part 4: API Testing Approach (1 hour)

1. API Test Scenarios

A. CRUD Operations Testing

Test Case ID	Description	Test Steps	Expected Result	Priority
API-001	Create VPC (POST /v1/vpcs)	1. Send POST request with valid payload (name, cidr_block, region, etc.). 2. Verify response status (201 Created). 3. Check response contains vpc_id and matches input.	VPC is created successfully with correct details.	High
API-002	List VPCs (GET /v1/vpcs)	1. Create a VPC. 2. Send GET request. 3. Verify response (200 OK) and that the created VPC appears.	List includes the newly created VPC.	High
API-003	Get VPC Details (GET /v1/vpcs/{vpc_id})	1. Create a VPC. 2. Retrieve it using vpc_id. 3. Verify response (200	VPC details match creation parameters.	High

		OK) and correct details.		
API-004	Update VPC (PUT /v1/vpcs/{vpc_id})	1. Create a VPC. 2. Modify DNS settings via PUT. 3. Verify response (200 OK) and updated values.	VPC attributes are updated successfully.	Medium
API-005	Delete VPC (DELETE /v1/vpcs/{vpc_id})	1. Create a VPC. 2. Delete it. 3. Verify response (204 No Content). 4. Confirm it no longer appears in GET list.	VPC is deleted and no longer accessible.	High

B. Input Validation

Test Case ID	Description	Test Steps	Expected Result	Priority
API-006	Invalid CIDR Block (POST /v1/vpcs)	1. Send POST with cidr_block="10.0.0.0/15". 2. Verify response (400 Bad Request).	API rejects invalid CIDR with error message.	High
API-007	Missing Required Field (POST /v1/vpcs)	1. Send POST without cidr_block. 2. Verify response (400 Bad Request).	API enforces required fields.	High

API-008	Invalid VPC Name (POST /v1/vpcs)	1. Send POST with name="test@vpc" (invalid chars). 2. Verify response (400 Bad Request).	API rejects invalid naming.	Medium

C. Authentication & Authorization

Test Case ID	Description	Test Steps	Expected Result	Priority
API-009	Unauthenticated Access (GET /v1/vpcs)	1. Send GET without API key/token. 2. Verify response (401 Unauthorized).	API blocks unauthorized access.	High
API-010	Cross-Account Access (GET /v1/vpcs/{vpc_id})	1. User A creates VPC. 2. User B tries to access it. 3. Verify response (403 Forbidden).	API enforces ownership.	High

D. Error Handling

Test Case ID	Description	Test Steps	Expected Result	Priority
API-011	Non-Existent VPC (GET /v1/vpcs/{vpc_id})	1. Send GET with invalid vpc_id. 2. Verify response (404 Not Found).	API returns proper "not found" error.	Medium

API-012	Duplicate CIDR (POST /v1/vpcs)	1. Create VPC with 10.0.0.0/16. 2. Try creating another with same CIDR. 3. Verify response (409 Conflict).	API prevents CIDR overlap.	High

E. Rate Limiting

Test Case ID	Description	Test Steps	Expected Result	Priority
API-013	API Throttling (POST /v1/vpcs)	1. Send 100+ requests in 1 minute. 2. Verify response (429 Too Many Requests).	API enforces rate limits.	Medium

F. Concurrent Request Handling

Test Case ID	Description	Test Steps	Expected Result	Priority
API-014	Parallel VPC Creation (POST /v1/vpcs)	1. Send 10 concurrent requests with unique CIDRs. 2. Verify all complete (201 Created). 3. Confirm no overlaps.	System handles concurrency correctly.	High

2. Test Implementation Plan

A. Testing Tools/Framework Recommendation

- Manual Testing: Postman (for exploratory testing)
- Automation: pytest + requests (Python) or RestAssured (Java)
- Load Testing: JMeter or Locust (for rate limiting/concurrency tests)

B. Test Data Setup Requirements

- Test Accounts: Multiple users with API keys.
- Test Regions: us-east-1, us-west-2.
- Test CIDRs: Valid (10.0.0.0/16, 192.168.0.0/24) and invalid (10.0.0.0/15).

C. Expected Response Codes & Payloads

Endpoint	Success	Error Cases
POST /v1/vpcs	201 Created	400 (Bad Input), 401 (Unauthorized), 409 (Conflict)
GET /v1/vpcs	200 OK	401 (Unauthorized)
GET /v1/vpcs/{vpc_id}	200 OK	404 (Not Found)
PUT /v1/vpcs/{vpc_id}	200 OK	403 (Forbidden), 404 (Not Found)
DELETE /v1/vpcs/{vpc_id}	204 No Content	404 (Not Found)

D. Performance Baseline Suggestions

Reads (GET): Under 500ms, Writes (POST/PUT/DELETE): Under 1 second.

Low Errors: <1% errors normally, <5%

Concurrency: Works with 10+ users acting simultaneously.

Goal: Ensure the API is fast, reliable, and secure under real-world use.

Candidate Name: Abdallah Assi

Email: assiabdallah@outlook.com

Phone: 0528686633

Time Spent

Section	Time spent
Part 1: Test Case Design	2 hours
Part 2: Bug Investigation	1 hour
Part 3: End-to-End Testing Strategy	1.5 hours
Part 4: API Testing Approach	1 hour
Total	5.5 hours