🏦 Pollinate Platform Engineering Technical Assessment

Project: Vendor Payment Risk Scoring Integration

Scenario

You are a Platform Engineer at **FinSure Capital**, a mid-sized FinTech company providing SME lending solutions.

FinSure is integrating with a third-party vendor called **RiskShield API**, which provides fraud and risk validation via REST API.

Before approving a loan, FinSure must call RiskShield's API to validate the applicant's identity and return a risk score.

Your task is to design, provision, containerise, and deploy a secure, production-ready integration platform on **Microsoft Azure**.

🎯 Objective

Build a cloud-native, secure integration service that:

1. Accepts a POST request containing applicant details
2. Calls RiskShield's REST API
3. Returns the vendor's risk score response
4. Is deployed to Azure using Terraform and Azure DevOps
5. Follows DevOps and security best practices

📦 Technical Requirements

1️⃣ Application Layer

Build a lightweight REST API service:

Endpoint:

POST /validate

Request:

```
{
  "firstName": "Jane",
  "lastName": "Doe",
  "idNumber": "9001011234088"
}
```

Behavior:

- Calls RiskShield API:

- POST https://api.riskshield.com/v1/score
- Uses API Key authentication (stored securely)
- Returns:

```
{
  "riskScore": 72,
  "riskLevel": "MEDIUM"
}
```

Requirements:

- Written in any language (Node.js, Python, GoLang , Java or .NET preferred)
- Proper error handling
- Logging
- Timeout handling
- Retry logic
- Correlation IDs

2️⃣ Containerisation

- Create a production-ready Dockerfile
- Use multi-stage builds
- Non-root user
- Small base image (e.g., alpine/distroless)
- Expose port correctly
- Include healthcheck

Demonstrate:

- Ability to optimise Docker images
- Security awareness

3️⃣ Infrastructure as Code (Terraform)

Provision infrastructure in Azure using Terraform.

Minimum expected resources:

- Resource Group
- Azure Container App OR Azure App Service

- Azure Container Registry (ACR)
- Azure Key Vault
- Azure Log Analytics Workspace
- Application Insights
- Managed Identity
- Role assignments

Infrastructure must:

- Use remote state (Azure Storage backend)
- Use modules
- Be reusable (dev/prod environments)
- Follow naming conventions
- Avoid hardcoded secrets

4️⃣ Security Requirements

The solution must:

- Store vendor API key in Azure Key Vault
- Use Managed Identity to retrieve secrets
- Restrict public exposure appropriately
- Enable HTTPS only
- Enable diagnostic logging
- Include basic threat modelling explanation

Bonus:

- Private endpoints
- Azure AD authentication
- Network restrictions

5️⃣ CI/CD Pipeline (Azure DevOps)

Create a YAML pipeline in:

Azure DevOps

Pipeline must:

Stage 1: Build

- Run tests
- Build Docker image
- Scan image (optional bonus)
- Push to ACR

Stage 2: Infrastructure

- Terraform init/plan
- Terraform apply (manual approval for prod)

Stage 3: Deploy

- Deploy container to Azure
- Smoke test endpoint

Must demonstrate:

- Use of service connections
- Variable groups
- Secure secret handling
- Separate environments (dev/prod)

📊 What This Assessment Evaluates

Skill Area	What We Look For
Azure	Proper use of PaaS services, identity, logging
Terraform	Modular, reusable, secure IaC
REST API	Proper HTTP handling, resilience
Docker	Optimisation and security
DevOps	Pipeline structure and environment strategy
Security	Secrets, identity, network boundaries
Observability	Logs, metrics, tracing

📁 **Expected Deliverables**

- Git repository with:
    - /app
    - /terraform
    - /pipelines
    - README.md (architecture explanation)
- Architecture diagram
- Instructions to run locally
- Instructions to deploy
- Security considerations
- Trade-offs explained

⏱ **Suggested Timebox**

6–10 hours (depending on depth)