Full Stack Engineer Assessment

Finconecta

OVERVIEW & PURPOSE

This assessment aims to evaluate the candidate's end-to-end development capabilities across both frontend and backend. It covers skills in Java Spring, React, MongoDB, PostgreSQL, AWS, and Kubernetes, with a focus on real-world architectural thinking, creativity, and technical depth.

INSTRUCTIONS

- You will be completing both coding and theoretical exercises.
- Please submit your deliverables via a GitHub, GitLab, or similar repository.
- You have 1 week to complete the assessment.
- Bonus sections are optional but highly encouraged.

Part 1: Full Stack Application

1. Backend Development (Java Spring)

- Build a Java Spring Boot application that includes full CRUD operations for a basic entity (e.g., User, Product, or similar).
- Use both MongoDB and PostgreSQL as databases. Explain how you manage persistence for each.
- Expose RESTful APIs using Spring Web or Spring WebFlux (bonus for reactive programming).
- Implement authentication using JWT.

2. Frontend Development (React)

Create a responsive React application with:

- A header, footer, and navigation bar (with at least 4 links to different sections/pages).
- At least two pages using React Router.
- A component that fetches data from the backend API and displays it.
- A form component to submit new data to the backend.
- Real-time updates to a list of items (bonus for using Socket.io).

Use React Hooks for state management.

Use CSS animations for hover effects on the navigation bar.

Apply a custom design theme using CSS.

Part 2: Infrastructure & DevOps

3. AWS Infrastructure

Write a CloudFormation template (or Terraform, if preferred) that provisions infrastructure to run the Spring Boot backend:

- EC2 for app hosting
- RDS for PostgreSQL
- MongoDB Atlas (or self-hosted)
- Apply security best practices (e.g., IAM roles, VPC, SGs)

4. Kubernetes Deployment (Bonus)

- Containerize the backend and frontend applications.
- Write Kubernetes YAML manifests for:
    - Deployments
    - Services
    - Ingress
- Show how you manage:
    - Secrets (e.g., DB credentials)
    - Database connections
    - Scalability and resilience

Part 3: Theoretical Questions

Java Spring Framework

- What is Dependency Injection, and why is it important?
- What's the difference between Spring MVC and Spring Boot?

Databases

- Compare MongoDB vs. PostgreSQL: data model, queries, scalability.
- When would you choose one over the other?

AWS

- Compare EC2 and ECS.
- How would you ensure high availability and fault tolerance?

Microservices & Kubernetes

- Define microservices and their benefits.
- Explain how Kubernetes supports microservices architectures.

Bonus (Optional)

- Implement Redux for global state management in the React app.
- Add a real-time chat feature using Socket.io.
- Implement Server-Side Rendering (SSR) for the React app using Next.js or similar.

DELIVERABLES

A single GitHub/GitLab repository or separate ones for frontend/backend (with clear instructions).