# Prueba – Ingeniero de Desarrollo JAVA (CredibanCo)

Se requiere crear una aplicación para administración de tarjetas de crédito y transacciones de compra, para esto se debe crear un API Restful para la creación y actualización de objetos y una interfaz gráfica para consultar los mismos.

**Generalidades**

* Usar lenguaje Java.
* Usar el framework Springboot o Kotlin para backend.
* Usar la tecnología de preferencia para el frontend (se sugiere Angular JS).

**Tiempo**

24 horas.

**Entregables**

* Link del código en un repositorio público (Se sugiere GitHub).
* Entrevista: Reunión en la cual se validará que el programa ejecute y funcione acorde a lo especificado, adicional se solicitará realizar algún cambio en el código. (1 hora aprox.)

**Proyecto 1 – Backend:**

Realizar un API Restful con las siguientes operaciones:

* Crear tarjeta: Permite crear una tarjeta en el sistema, debe retornar un número al azar entre 1 y 100 que va a hacer un **número de validación** para poder enrolar la tarjeta (siguiente método) y debe almacenarse con un estado ‘Creada’.

**Datos request:** PAN (Número de la tarjeta – 16 a 19 dígitos), titular, cédula (10 a 15 caracteres), tipo: (Crédito, Débito) y teléfono (10 dígitos)

**Datos Response**: Código de respuesta (00, 01), mensaje (Éxito, Fallido), número de validación, PAN enmascarado (solo deben poder ser visibles los primeros 6 dígitos y los últimos 4, los demás deben ser caracteres de relleno, por ejemplo: 123456\*\*\*\*3456)

* Enrolar Tarjeta: Permite activar la tarjeta para poder realizar una transacción. Cambia el estado a ‘Enrolada’ solo si concuerda el **número de validación** generado al momento de la creación.

**Datos request:** PAN, numero de validación

**Datos response:** Código de respuesta (00, 01, 02), mensaje (Éxito, Tarjeta no existe, Número de validación inválido), PAN enmascarado

* Consultar Tarjeta: permite traer los datos de una tarjeta.

**Query Param:** PAN

**Datos response:** PAN enmascarado, titular, cédula, teléfono y estado

* Eliminar Tarjeta: Permite borrar una tarjeta del sistema

**Datos request:** PAN, numero de validación, PAN

**Datos response:** Código de respuesta (00, 01), mensaje (Se ha eliminado la tarjeta, No se ha eliminado la tarjeta)

* Crear Transacción: Permite aprobar una compra cuando la tarjeta utilizada existe y su estado es ‘Enrolada’

**Datos request:** PAN, número de referencia (6 dígitos), total compra, dirección de compra

**Datos response:** Código de respuesta (00, 01, 02), mensaje (Compra exitosa, Tarjeta no existe, Tarjeta no enrolada), estado transacción (Aprobada, rechazada), número de referencia

* Anular Transacción: Permite anular una compra solo si se hizo hace menos de 5 minutos

**Datos request:** PAN, número de referencia, total compra

**Datos response:** Código de respuesta (00, 01, 02), mensaje (Compra anulada, numero de referencia inválido, No se puede anular transacción), número de referencia

**Proyecto 2 - Frontend:**

Se requiere una pantalla web para:

* Visualizar: Datos de tarjetas (Debe mostrar el PAN enmascarado)
* Visualizar: Datos de transacciones

**Aspectos técnicos:**

* Usar spring-framework (spring-boot, mvc).
* Usar javax.validation.constraints.
* Tener en cuenta la estructura de paquetes.
  + Entidades: com.credibanco.assessment.card.model
  + DTOs: com.credibanco.assessment.card.dto
  + Servicios: com.credibanco.assessment.card.service
  + Servicios (implementación): com.credibanco.assessment.card.service.impl
  + Controladores: com.credibanco.assessment.card.controller
  + Cliente (API): com.credibanco.assessment.card.api.client
  + Excepciones: com.credibanco.assessment.card.exceptions
  + Excepciones (cliente): com.credibanco.assessment.card.api.client.exceptions
* Las entidades y DTOs deben estar en artefactos independientes. Es decir, los objetos de request deben ser diferentes a los objetos de base de datos.
* Incluir actuadores.
* Usar base de datos Oracle o MySQL.
* Hacer buen uso de los métodos **HTTP**: GET, POST, PUT, DELETE.
* Hacer buen uso de los códigos **HTTP**: 200, 201, 404, 500.