

Colegio Universitario de Cartago

Primer Proyecto Programado

Sistema de Tarjetas

El banco ABC requiere de renovar su sistema de procesamiento de transacciones de clientes con tarjetas de débito y crédito para brindar un mejor servicio a sus clientes y agilizar los movimientos que estos realizan.

Este sistema está formado por varios componentes que se interconectan entre sí, de tal forma que su interacción permite de manera fluida que los clientes realicen sus transacciones.

Durante el desarrollo de este proyecto, su equipo de trabajo construirá diferentes componentes de sistema de tarjetas, entregando finalmente en tres iteraciones el producto final.

Primer alcance - Sistema base, cajeros automáticos.

Para la primera parte del proyecto se debe trabajar en los siguientes elementos:

1. Base de datos del sistema de tarjetas (modelo relacional en MySQL).
2. Base de datos del Core Bancario (Modelo relacional en SQL Server).
3. Autorizador de transacciones.
4. Componente del Core Bancario.
5. Simulador de cajero automático.

La idea es que los modelos de base de datos propuestos para cada uno de los sistemas mencionados incorporen los diferentes elementos que atiendan exclusivamente las historias de usuario a atender. Los datos que se agreguen a estas bases de datos inicialmente se incorporarán mediante scripts.

A continuación, se detallan las historias de usuario y los criterios de aceptación que las acompañan.

| | |
|---|--|
| ID | AUT1 |
| Rol | Dueño del sistema |
| Objetivo | Autorizar retiros de tarjeta |
| Para | Atender las solicitudes de los cajeros automáticos |
| Criterios de aceptación | |
| <ol style="list-style-type: none"> 1. El autorizador debe recibir una trama en formato XML o JSON que incorpore la siguiente información: <ol style="list-style-type: none"> a. Número de tarjeta (debe enviarse y almacenarse cifrado). b. PIN (debe enviarse y almacenarse cifrado). c. Fecha de vencimiento (debe enviarse y almacenarse cifrado). d. Código de verificación (debe enviarse y almacenarse cifrado). e. Identificación del cajero. f. Tipo de transacción (Ejemplo: retiro, consulta). g. Monto de la transacción. 2. Deben realizarse las siguientes validaciones contra la base de datos del sistema de tarjetas: <ol style="list-style-type: none"> a. Todos los datos son obligatorios. b. El monto solo es obligatorio si el tipo de transacción es retiro. c. El número de tarjeta debe existir y corresponder a una tarjeta activa. d. El PIN debe ser el correcto de la tarjeta. e. La fecha de vencimiento debe ser la asociada a la tarjeta y debe estar vigente. f. El código de verificación debe ser el asociado a la tarjeta. g. El código de identificación del cajero automático debe existir y ser válido. 3. Al recibir una transacción si todas las validaciones anteriores son correctas entonces: <ol style="list-style-type: none"> a. Si el tipo de transacción es retiro entonces: | |

- i. Si la tarjeta es de débito se debe de verificar contra el Core Bancario si el cliente tiene el suficiente dinero para retirar el monto solicitado, si lo tiene se autoriza el retiro. Ver HU Core1.
 - ii. Si es de crédito, se debe verificar el monto disponible para adelantos de efectivo en el sistema de tarjetas y si se puede retirar, entonces se autoriza el retiro.
4. Si el tipo de transacción es retiro entonces:
 - a. Si el retiro es autorizado, debe responder con un “OK”, dentro de una trama XML o JSON, junto con el código de autorización (número aleatorio de 8 dígitos que no se puede repetir en el mismo día). Ejemplo: {“status”: “OK”, “autorización”: 12345678}.
 - b. Si el retiro no es autorizado, entonces debe indicar alguno de los siguientes códigos de motivo de acuerdo con lo sucedido, dentro de una trama XML o JSON:
 - i. 1: Fondos insuficientes (en caso de obtener el valor “INSUF” del Core).
 - ii. 2: Datos incorrectos.
 - iii. 3: Tarjeta inactiva.
 - iv. 4: Tarjeta vencida.
 - v. 5: Error no controlado.

Anotaciones técnicas:

- El autorizador debe implementarse usando sockets síncronos. Puede atender múltiples solicitudes de cajeros diferentes a la vez.
- El puerto donde ejecuta debe poder indicarse en configuración.
- El algoritmo de cifrado de los datos sensibles, debe ser AES.
- Considerar en su solución que los números de cuenta debe tener un tamaño fijo.

| | |
|-----|-------------------|
| ID | AUT2 |
| Rol | Dueño del sistema |

| | |
|--|--|
| Objetivo | Autorizar consultas de tarjeta débito |
| Para | Atender las solicitudes de los cajeros automáticos |
| Criterios de aceptación | |
| <ol style="list-style-type: none"> 1. El autorizador debe recibir una trama en formato XML o JSON que incorpore la siguiente información: <ol style="list-style-type: none"> a. Número de tarjeta (debe enviarse y almacenarse cifrado). b. PIN (debe enviarse y almacenarse cifrado). c. Fecha de vencimiento (debe enviarse y almacenarse cifrado). d. Código de verificación (debe enviarse y almacenarse cifrado). e. Identificación del cajero. f. Tipo de transacción (Ejemplo: retiro, consulta). 2. Deben realizarse las siguientes validaciones contra la base de datos del sistema de tarjetas: <ol style="list-style-type: none"> a. Todos los datos son obligatorios. b. El número de tarjeta debe existir y corresponder a una tarjeta activa. c. El PIN debe ser el correcto de la tarjeta. d. La fecha de vencimiento debe ser la asociada a la tarjeta y debe estar vigente. e. El código de verificación debe ser el asociado a la tarjeta. f. El código de identificación del cajero automático debe existir y ser válido. 3. Al recibir una transacción si todas las validaciones anteriores son correctas entonces: <ol style="list-style-type: none"> a. Si el tipo de transacción es consulta, entonces: <ol style="list-style-type: none"> i. Si la tarjeta es de débito entonces se debe obtener del Core Bancario el monto del saldo de la cuenta, asociado al número de tarjeta que se está utilizando. Ver HU Core1. ii. Si la tarjeta es de crédito, se debe obtener del sistema de tarjetas el monto del disponible para avances de efectivo, asociado a la tarjeta que se está utilizando. | |

4. Si el tipo de transacción es consulta entonces:
- Si al consultar el Core o al obtener el monto de adelanto de efectivo se obtuvo una respuesta exitosa se debe enviar en una trama XML o JSON el monto formateado con separador de miles y punto decimal.
 - Si la respuesta no es exitosa, se debe responder el código 5, al igual que en la trama de retiro.

Anotaciones técnicas:

- El autorizador debe implementarse usando sockets síncronos. Puede atender múltiples solicitudes de cajeros diferentes a la vez.
- El puerto donde ejecuta debe poder indicarse en configuración.
- El algoritmo de cifrado de los datos sensibles, debe ser AES.
- Considerar en su solución que los números de cuenta debe tener un tamaño fijo.

| | |
|---|-------------------------------|
| ID | CORE1 |
| Rol | Dueño del sistema |
| Objetivo | Verificar el saldo en el Core |
| Para | Autorizar transacciones |
| Criterios de aceptación | |
| 1. El Core debe recibir una trama en formato de texto plano proveniente del autorizador que contendrá los datos que se requieren para verificar si el cliente tiene los fondos suficientes para realizar un retiro o bien, para consultar su saldo. | |
| 2. La trama debe incluir: | |
| Tipo de transacción (ocupa un espacio) | 1: Retiro 2: Consulta |
| Número de cuenta a consulta (ocupa el tamaño acorde a la definición del número de cuenta que realizó en su solución) | Ejemplo: 45678909-3 |
| Número de tarjeta (19 espacios) | Ejemplo: 4567 89** **** 1234 |

| | |
|---|-------------------|
| El tercer grupo de números siempre debe ser asteriscos. | |
| <p>Monto</p> <p>El monto ocupará 8 espacios, incluyendo dos decimales, siempre los últimos dos dígitos son decimales.</p> | Ejemplo: 15000000 |

3. Al recibir los datos de la trama, si el tipo de transacción es retiro entonces:

- Si hay fondos suficientes en el saldo de la cuenta se debe responder “OK”.
- Si no hay fondos suficientes se debe responder “INSUF”.
- Cualquier otra situación se debe responder “ERROR”.

4. Si el tipo de transacción es consulta entonces:

- En caso de éxito se debe responder “OK” junto con el saldo de la cuenta, el saldo se debe representar en 19 espacios, incluyendo decimales, no es necesario incluir el punto decimal, se asume que los dos últimos dígitos son los decimales. Para completar los 19 espacios se debe rellenar con ceros a la izquierda. Ejemplo: 00000000000097654.25.
- Cualquier otra situación se debe responder “ERROR”.

| | |
|--|---------------------------------------|
| ID | AUT3 |
| Rol | Dueño del sistema |
| Objetivo | Cambiar el PIN de las tarjetas |
| Para | Facilitar el cambio desde los cajeros |
| Criterios de aceptación | |
| <p>1. El autorizador debe recibir una trama en formato XML o JSON que incorpore la siguiente información:</p> <ol style="list-style-type: none"> Número de tarjeta (debe enviarse y almacenarse cifrado). PIN actual (debe enviarse y almacenarse cifrado). PIN nuevo (debe enviarse y almacenarse cifrado) | |

- d. Fecha de vencimiento (debe enviarse y almacenarse cifrado).
 - e. Código de verificación (debe enviarse y almacenarse cifrado).
 - f. Identificación del cajero.
 - g. Tipo de transacción (Ejemplo: retiro, consulta, cambio de pin).
2. Deben realizarse las siguientes validaciones contra la base de datos del sistema de tarjetas:
 - a. Todos los datos son obligatorios.
 - b. El número de tarjeta debe existir y corresponder a una tarjeta activa.
 - c. El PIN actual debe ser el correcto de la tarjeta.
 - d. La fecha de vencimiento debe ser la asociada a la tarjeta y debe estar vigente.
 - e. El código de verificación debe ser el asociado a la tarjeta.
 - f. El código de identificación del cajero automático debe existir y ser válido.
3. Al recibir una transacción si todas las validaciones anteriores son correctas entonces:
 - a. Si el tipo de transacción es cambio de Pin, entonces:
 - i. Se debe actualizar en el sistema de tarjetas el pin asociado a la tarjeta.
 - ii. El nuevo PIN se debe almacenar cifrado.
 - iii. En caso de éxito se debe responder "OK".
 - iv. Cualquier otra situación se debe responder "ERROR".

Anotaciones técnicas:

- El autorizador debe implementarse usando sockets síncronos. Puede atender múltiples solicitudes de cajeros diferentes a la vez.
- El puerto donde ejecuta debe poder indicarse en configuración.
- El algoritmo de cifrado de los datos sensibles, debe ser AES.

| | |
|--|---|
| ID | AUT4 |
| Rol | Dueño del sistema |
| Objetivo | Registrar bitácoras |
| Para | Dejar rastros de auditoría de las operaciones |
| Criterios de aceptación | |
| <ol style="list-style-type: none">1. El autorizador debe llevar una bitácora de operaciones que se almacenará en el servidor en un archivo de texto.2. Los datos que se escribirán en la bitácora serán:<ol style="list-style-type: none">a. Fecha de bitácora.b. Número de tarjeta (debe escribirse enmascarado de la siguiente forma: 1345 45** **** 2587).c. Identificación del cajero.d. Identificación del clientee. Tipo de transacción (Ejemplo: retiro, consulta, cambio pin). Se debe registrar su nombre.f. Monto de la transacción (si aplica).3. El formato del registro de estos datos de la bitácora debe ser JSON y almacenado así, por ejemplo: 15/01/2024: {"tarjeta": "1345 45** **** 2587", "cajero": 1509, "cliente": "112340456", "tipo": "Retiro", "Monto": "75000.00"}4. El registro de bitácora debe realizarse en segundo plano en un hilo independiente y no debe interrumpir la ejecución de las operaciones del autorizador. Las solicitudes de escritura en bitácora deben "encolarse", para poder acceder al archivo de bitácora de forma ordenada y evitar bloqueos.5. Se debe registrar bitácora para todo tipo de transacción. | |

| | |
|---|---|
| ID | AUT5 |
| Rol | Dueño del sistema |
| Objetivo | Confirmar retiros |
| Para | Aplicar el movimiento realizado de retiro |
| Criterios de aceptación | |
| <ol style="list-style-type: none"> 1. El autorizador debe recibir una trama en formato XML o JSON que contenga una confirmación de parte de los cajeros automáticos de que se realizó un retiro de dinero de manera exitosa. Esta trama debe incluir: <ol style="list-style-type: none"> a. Código de autorización. b. Número de tarjeta (debe enviarse y almacenarse cifrado). c. Fecha de vencimiento (debe enviarse y almacenarse cifrado). d. Código de verificación (debe enviarse y almacenarse cifrado). e. Identificación del cajero. f. Tipo de transacción (Ejemplo: confirmación). g. Monto de la transacción. 2. Deben realizarse las siguientes validaciones contra la base de datos del sistema de tarjetas: <ol style="list-style-type: none"> a. Todos los datos son obligatorios. b. El número de tarjeta debe existir y corresponder a una tarjeta activa. c. La fecha de vencimiento debe ser la asociada a la tarjeta y debe estar vigente. d. El código de verificación debe ser el asociado a la tarjeta. e. El código de identificación del cajero automático debe existir y ser válido. f. El código de autorización debe existir para la tarjeta indicada. 3. Una vez se reciba esta confirmación y si las validaciones son correctas, si la tarjeta es de débito: | |

- a. Entonces se debe enviar la transacción al Core para que registre el movimiento en la cuenta del cliente y responder nuevamente “OK” junto con el código de autorización. Ver HU Core2
 - b. Si el Core respondiera con un error, se debe responder “ERROR”.
4. Si la tarjeta es de crédito, el movimiento no se envía a las cuentas del cliente, sino que se realiza un registro del movimiento en los movimientos de la tarjeta de crédito del cliente, del sistema de tarjetas y se coloca en estado “Pendiente”.

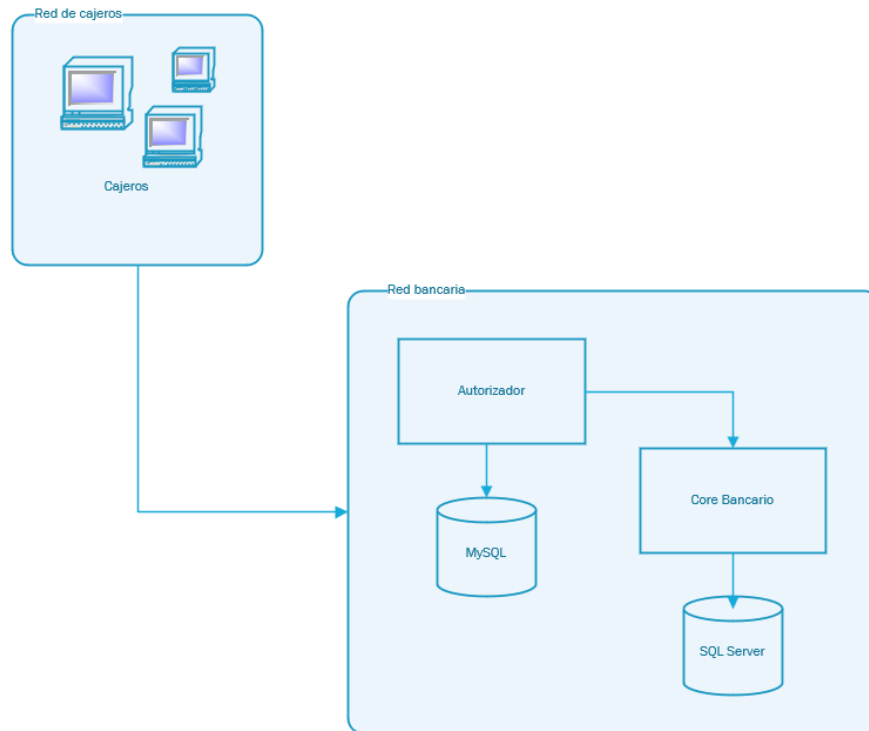
| | |
|--|----------------------------------|
| ID | CORE2 |
| Rol | Dueño del sistema |
| Objetivo | Registrar movimientos |
| Para | Rebajar el dinero de los retiros |
| Criterios de aceptación | |
| 1. El Core debe recibir una trama en formato de texto plano proveniente del autorizador que contendrá los datos que se requieren para registrar los retiros de dinero del cliente. | |
| 2. La trama debe incluir: | |
| Tipo de transacción (ocupa un espacio) | 1: Retiro 2: Consulta |
| Número de cuenta a registrar el movimiento (ocupa el tamaño acorde a la definición del número de cuenta que realizó en su solución) | Ejemplo: 45678909-3 |
| Número de tarjeta (19 espacios) El tercer grupo de números siempre debe ser asteriscos. | Ejemplo: 4567 89** **** 1234 |
| Código de autorización | Ejemplo: 12345678 |
| Monto | Ejemplo: 15000000 |

| | |
|--|---|
| El monto ocupará 8 espacios, incluyendo dos decimales, siempre los últimos dos dígitos son decimales. | |
| <p>3. Con estos datos debe registrar el movimiento en la cuenta del cliente y en la descripción del detalle del movimiento indicar: Retiro # Tarjera y Código autorización.</p> <p>4. Si la operación es exitosa debe responder “OK”, de lo contrario debe responder “ERROR”.</p> | |
| ID | CORE3 |
| Rol | Dueño del sistema |
| Objetivo | Registrar bitácoras |
| Para | Dejar rastros de auditoría de las operaciones |
| Criterios de aceptación | |
| <p>1. El Core debe llevar una bitácora de operaciones que se almacenará en el servidor en un archivo de texto.</p> <p>2. Los datos que se escribirán en la bitácora serán los recibidos en la trama, siguiendo un formato JSON. Debe indicar la fecha de la bitácora y también el resultado de la operación, es decir, respuesta.</p> <p>3. El formato del registro de estos datos de la bitácora debe ser JSON y almacenado así, por ejemplo:</p> <p>15/01/2024: {"tarjeta": "1345 45** **** 2587", "cuenta": "45678909-3", "código de autorización": "11234045", "tipo": "Retiro", "Monto": "07500000"}</p> <p>4. El registro de bitácora debe realizarse en segundo plano en un hilo independiente y no debe interrumpir la ejecución de las operaciones del core. Las solicitudes de escritura en bitácora deben “encolarse”, para poder acceder al archivo de bitácora de forma ordenada y evitar bloqueos.</p> <p>5. Se debe registrar bitácora para todo tipo de transacción.</p> | |

| | |
|--|--|
| ID | SIM1 |
| Rol | Dueño del sistema |
| Objetivo | Un simulador de cajero automático |
| Para | Corroborar la funcionalidad de las transacciones del autorizador |
| Criterios de aceptación | |
| <ol style="list-style-type: none">1. Debe crear una interfaz que simulará la funcionalidad de un cajero automático2. para poder realizar cada una de las transacciones: retiro (+ confirmación), consulta y cambio de pin.3. Cada una de las transacciones debe tener una pantalla independiente.4. Cada transacción capturará los datos que solicita únicamente. No es necesario validarlos, el autorizador realizará la validación.5. Cada interfaz debe armar la trama que requiere el autorizador y enviársela y mostrar el resultado de realizar la operación.6. Recuerde que los datos sensibles deben transmitirse cifrados.7. En el caso del retiro, hay que considerar que consta de dos pasos, primero la solicitud del retiro y posteriormente la confirmación. | |

Aspectos técnicos generales obligatorios por cumplir

1. El simulador debe desarrollarse en C#.
2. El socket del autorizador debe realizarlo en Python BD MySQL.
3. El socket del Core debe realizarlo en Java BD SQL Server.

Diagrama general de arquitectura (Alto nivel)**Entregables**

A continuación, se lista lo que debe entregar en el proyecto.

1. Documentación de análisis y diseño.
 - a. Portada.
 - b. Introducción (Resumen del problema a resolver).
 - c. Diagrama de base de datos (presentar propuestas apenas lo tenga definido).
 - d. Diagramas de casos de uso.
 - e. Diagramas de clases.
 - f. Conclusiones y recomendaciones.
 - g. Bibliografía. (Formato APA7)
2. Implementación de bases de datos.
3. Autorizador de transacciones. (Código fuente)

4. Componente del Core Bancario. (Código fuente)
5. Simulador de cajero automático. (Código fuente)

Aspectos administrativos

- El proyecto debe desarrollarse en los equipos de trabajo definidos.
- Corresponde desarrollar 3 historias por integrante. Se desarrollan de forma individual, no se comparten. Indicar en el canal de TEAMS, la próxima semana a más tardar la forma de distribución de las historias.
- La fecha de entrega es el 11/2/2025 antes de las 6 p.m. La presentación del proyecto será ese día a partir de esa hora, deben estar presentes todos los miembros del grupo y sin programar.
- Si se llega a detectar entrega tardía de lo solicitado (después de las 6 p.m.), automáticamente el proyecto será calificado base 90.
- El valor de este avance es del 15%.
- Si bien las historias de usuario se atienden en equipo, la revisión es individual, por lo que cada persona es responsable de sus criterios de aceptación desarrollados.
- La calificación de cada proyecto, se puede desarrollar de forma individual (historias de cada uno) o grupal. Esto se solicita el día en que van a realizar la entrega.
- Si presentan funcionalidad pero no presentan el código, dicha entrega no será tomada en cuenta como válida perdiendo el porcentaje respectivo. Así también si el código no coincide con lo mostrado o si no respetan la tecnología e indicaciones brindadas.
- Durante la presentación, aunque son componentes distintos deben estar comunicados entre sí como un proyecto integrado, esto es que pueden presentarse en equipos individuales o en un solo equipo, pero deben funcionar como un conjunto de aplicaciones integradas, en caso de presentar de forma

aislada algún componente, esta revisión será sancionada y calificada sobre base 85 en vez de base 100.