**Producto Académico n. ° 03 – Consolidado 2**

1. **Consideraciones**

|  |  |
| --- | --- |
| **Criterio** | **Detalle** |
| Temas | Programación visual y con eventos.  Conexión a base de datos. API JDBC. Consultas SQL.  Manipulación de Base de Datos. CRUD a tablas |
| Enunciado | Desarrollar una ***segunda versión del software*** que simula un ATM.  La interfaz de usuario de cajero automático contiene los siguientes componentes: una pantalla que muestra mensajes al usuario, un teclado que recibe datos numéricos de entrada del usuario, un dispensador de efectivo que dispensa efectivo al usuario, y una ranura de depósito que recibe dinero del usuario.    Esta segunda versión deberá implementar la siguiente funcionalidad:  Cuando el Cliente se acerca al ATM se visualiza:   1. La pantalla muestra un mensaje de bienvenida y pide al usuario que introduzca un número de cuenta. 2. El usuario introduce un número de cuenta de cinco dígitos, mediante el uso del teclado. 3. En la pantalla aparece un mensaje, en el que se pide al usuario que introduzca su NIP (número de identificación personal) asociado con el número de cuenta especificado. 4. El usuario introduce un NIP de cinco dígitos mediante el teclado numérico. 5. Si el usuario introduce un número de cuenta valido y el NIP correcto para esa cuenta, la pantalla muestra el menú principal:   1. Ver mi saldo  2. Retirar efectivo  3. Depositar dinero  4. Salir  Después el usuario elegirá si desea realizar una transacción (oprimiendo 1, 2 o 3) o salir del sistema (oprimiendo 4).  Si el usuario selecciona **opción 1** para solicitar su saldo, la pantalla mostrara el saldo de esa cuenta bancaria. Para ello, el ATM deberá obtener el saldo de una base de datos.  Si usuario elige la **opción 2** para hacer un retiro:   1. La pantalla muestra un menú que contiene montos de retiro estándar: S/ 20 (opción 1), S/ 40 (opción 2), S/ 60 (opción 3), S/ 100 (opción 4) y S/ 200 (opción 5). El menú también contiene una opción que permite al usuario cancelar la transacción. 2. El usuario introduce la selección del menú mediante el teclado numérico. Si el monto a retirar elegido es mayor que el saldo de la cuenta del usuario, la pantalla muestra un mensaje indicando esta situación y pide al usuario que seleccione un monto más pequeño. Entonces el ATM regresa al paso 1. 3. Si el monto a retirar elegido es menor o igual que el saldo de la cuenta del usuario (es decir, un monto de retiro aceptable), el ATM procede al paso 4. Si el usuario opta por cancelar la transacción (opción 6), el ATM muestra el menú principal y espera la entrada del usuario. 4. Si el dispensador contiene suficiente efectivo para satisfacer la solicitud, el ATM procede al paso 5. En caso contrario, la pantalla muestra un mensaje indicando el problema y pide al usuario que seleccione un monto de retiro más pequeño. Después el ATM regresa al paso 1. 5. El ATM carga el monto de retiro al saldo de la cuenta del usuario en la base de datos del banco (es decir, resta el monto de retiro al saldo de la cuenta del usuario). 6. El dispensador de efectivo entrega el monto deseado de dinero al usuario.   Si usuario elige la **opción 3** para hacer un depósito:   1. La pantalla muestra un mensaje que pide al usuario que introduzca un monto de depósito o que escriba 0 (cero) para cancelar la transacción. 2. El usuario introduce un monto de depósito o 0 mediante el teclado numérico. 3. Si el usuario especifica un monto a depositar, el ATM procede al paso 4. Si elige cancelar la transacción (escribiendo 0), el ATM muestra el menú principal y espera la entrada del usuario. 4. La pantalla muestra un mensaje indicando al usuario que introduzca un depósito en la ranura para depósitos. 5. Si la ranura de depósitos recibe el dinero dentro de un plazo de tiempo no mayor a 2 minutos, el ATM abona el monto del depósito al saldo de la cuenta del usuario en la base de datos (es decir, suma el monto del depósito al saldo de la cuenta del usuario).   Una vez que el sistema ejecuta una transacción en forma exitosa, debe volver a mostrar el menú principal para que el usuario pueda realizar transacciones adicionales. Si el usuario elije salir del sistema, la pantalla debe mostrar un mensaje de agradecimiento y después el mensaje de bienvenida para el siguiente usuario. |
| Instrucciones para empezar a elaborar el proyecto | * Forma grupos de 2 estudiantes para el desarrollo del producto académico * **Elabora un proyecto integrador** que contemple las siguientes partes:  1. **Diseño de Interfaces de Usuario**   Descripción de GUIs.   1. **Diseño de Clases**   Diagrama de clases.   1. **Diseño de Base Datos**   Diseño de Tablas.   1. **Construcción**   Código en Java.   * Incluye un archivo Word con el listado de nombres de los integrantes del grupo, nómbralo “Integrantes.docx”. Incluye el archivo dentro de la carpeta del proyecto. * Deberás subir el proyecto desarrollado al aula virtual de la asignatura, como archivo adjunto mediante el enlace de presentación. * Recuerda revisar **la rúbrica de evaluación** para conocer los criterios de evaluación |
| Referencias | Deitel, P. y Deitel, H. (2017). *Java How to program.* (11° ed.). New York, USA: Pearson |

1. **Rúbrica de evaluación:**

A continuación, se presenta la escala de valoración, en base a la cual se evaluará la participación en este foro, donde la escala máxima por categoría equivale a 5 puntos y la mínima 0.

| **Criterios** | **Sobresaliente**  **(5 puntos)** | **Suficiente**  **(3 puntos)** | **En proceso**  **(2 puntos)** | **En inicio**  **(0 puntos)** |
| --- | --- | --- | --- | --- |
| **Diseño de interfaces de usuarios** | Diseña GUIs y aplica correctamente la jerarquía de clases Swing en la construcción de Interfaces graficas de usuarios usando eventos. | Diseña GUIs y aplica adecuadamente la jerarquía de clases Swing en la construcción de Interfaces graficas de usuarios usando eventos. | Diseña GUIs y aplica parcialmente la jerarquía de clases Swing en la construcción de Interfaces graficas de usuarios usando eventos. | No aplica la jerarquía de clases Swing en la construcción de Interfaces graficas de usuarios usando eventos. |
| **Diseño de clases y objetos** | Diseña clases y define eficientemente atributos y métodos utilizando conceptos de la programación orientada a objetos, tales como encapsulamiento y herencia. | Diseña clases y define adecuadamente atributos y métodos utilizando conceptos de la programación orientada a objetos, tales como encapsulamiento y herencia. | Utiliza conceptos de la programación orientada a objetos de manera parcial en el diseño de clases y objetos. | No utilizando conceptos de la programación orientada a objetos, tales como encapsulamiento y herencia, en el diseño de clases y objetos. |
| **Diseño de Base de Datos** | Desarrolla la aplicación con acceso a base de datos utilizando las clases de JDBC de manera eficiente. | Desarrolla la aplicación con acceso a base de datos utilizando las clases de JDBC de manera adecuada. | Desarrolla la aplicación con acceso a base de datos de manera parcial. | No desarrolla la aplicación con acceso a base de datos y no utiliza las clases de JDBC. |
| **Construcción** | Construye clases y objetos usando eficientemente la Programación Orientada a objetos en Java. | Construye clases y objetos usando adecuadamente la Programación Orientada a objetos en Java. | Construye clases y objetos usando Java de manera parcial. | No construye clases y objetos usando Java. |
| **Total** |  |  |  |  |
| **Nota** |  | | | |