- Se requiere hacer un programa orientado a objetos basado en tablas relacionales y con arquitectura stand alone.
- En un kiosco hemos puesto una pantalla más amigable al usuario para que se puedan rentar las propiedades para grandes empresas.

## Qué se pide: (100%) Código java JDBC para acceder las tablas del modelo

Nota General: para las funcionalidades todos los datos se deben ir almacenando en memoria en el modelo unificado presentado en clase y siempre debe existir un botón guardar que tome toda la información en memoria y la almacene en las tablas.

Casos de Uso

### 1.CU Login (10 puntos)

-se valida un usuario y contraseña contra una tabla en la base de datos

-se debe desplegar un menú con las opciones de:

* listar propiedades del usuario conectado

* Reporte de rentas.

### 2.CU Listar propiedades (10 puntos)

-se deben consultar desde la base de datos las propiedades del usuario conectado

-se deben desplegar el listado de propiedades obtenido

-el usuario puede en esta pantalla seleccionar una propiedad y luego ir hacia la funcionalidad de 'Crear Renta' y/o de 'Listar Rentas'

### 3.CU Crear Renta (20 puntos)

-crear renta: incluido un pago con Tarjeta de Crédito (renta tiene id, la fecha actual, propiedad). Se debe totalizar y desplegar el total de la renta.

-agregar línea de servicio:

se debe desplegar una lista de servicios para que el usuario seleccione uno de ellos, la lista de servicios debe ser consultada desde la base de datos.

Se debe totalizar y desplegar el total de la renta.

A medida que se agregan los servicios se deben desplegar en una Tabla (Jtable)

Se debe totalizar y desplegar el total de la renta.

-eliminar línea de servicio:

Se debe seleccionar desde la tabla un servicio para eliminar, la tabla debe refrescarse.

Se debe totalizar y desplegar el total de la renta.

-modificar línea de servicio

Se debe seleccionar desde la tabla un servicio para modificar, se debe solicitar el nuevo número de inquilinos, la tabla debe refrescarse.

-existe un botón en la pantalla de crear renta que al usuarlo toma la información en memoria y la almacena en la base de datos.

Se debe totalizar y desplegar el total de la renta 'CU Totalizar Renta'.

-existe un botón para limpiar y crear una nueva renta.

## 4.CU Listar Rentas  (10 puntos)

-se introduce una fecha

-se despliega una lista de rentas de la fecha dada para la propiedad seleccionada, la lista debe ser consultada desde la base de datos.

## 5.CU Cargar renta desde la base de datos (10 puntos)

-el usuario selecciona una renta del listado de rentas por fecha (CU Listar Rentas) y se debe cargar desde la base de datos hacia los objetos en memoria la renta seleccionada: propiedad, renta, servicios, pago, ….

## 6. CU Modificar Renta (10 puntos)

-una vez la renta esté cargada en memoria se pueden: agregar líneas, eliminar líneas, modificar líneas; similar al CU de Crear Renta.

-existe un botón en esta pantalla que al usuarlo toma la información en memoria y la almacena (actualiza) en la base de datos.

## 7. CU Eliminar Renta (10 puntos)

-una vez la renta esté cargada en memoria se puede habilitar un botón para Eliminar la Renta de la base de datos.

## 8. CU Reporte de Rentas (20 puntos)

-se debe realizar una consulta a la base de datos para consultar para el usuario actual: para cada propiedad el total de las rentas por año; los resultados de la consulta se deben mostrar en pantalla. se debe desplegar el total general del reporte y debe ordenarse por año. Se debe retornar un DTO con la información necesaria para mostrar en una jtable.

**Notas:**

-se maneja una renta a la vez en memoria

Notas para representar el modelo de datos a partir del modelo de clases (parte I de la entrega)

A. Se debe revisar la capa de Entidades SAP
B. Las relaciones de composición son relaciones identificantes.
C. En la memoria no se montan todos los datos; en las tablas están todos los datos.
D. El modelo de clases no siempre es bidireccional, en la base de datos siempre se debe establecer la bidireccionalidad
E. El modelo de clases solo establece cardinalidad respecto a los datos que se cargan en memoria; en las tablas se debe establecer la cardinalidad pensando en todos los datos.
F. En el modelo de clases no siempre se reflejan las relaciones muchos a muchos; debido a la navegabilidad y cardinalidad representada en memoria a los datos cargados.

**Interfaz Gráfica de Usuario**

Usted debe proveer la interfaz gráfica del proyecto.

Si decide no hacer interfaz gráfica (perderá el 30% del puntaje en cada CU) debe proveer un main donde se pueda probar la capa de negocio y la de integración.

**Grupos**

La entrega se realizará en grupos de trabajo. Los grupos no podrán cambiar su conformación y desde el comienzo dichos grupos estarán identificados plenamente.

**Entregables**

- Archivo .zip con el código fuente de las clases
- Nombre de dos esquemas/usuarios de bases de datos donde se pueda verificar el código

**Observaciones**

- Se reducirán puntos por malas prácticas de programación: ○ código “quemado”. Por ejemplo, usar valores constantes en donde no se deba.
- El diagrama de clases y la implementación deben ser concordantes.
- Si no hay código, la nota corresponderá a 0.0
- SUSTENTACION INDIVIDUAL: en caso de no ser exitosa la sustentación, se reconocerá el 20% del total obtenido.
- Cada clase deberá tener el nombre completo de los autores

**Restricciones**

- La lógica y la presentación deben estar separadas.
- Se deben leer datos en la presentación y procesarlos en la lógica de negocio
    - ○ Toda la creación y procesamiento de objetos debe realizarse en la lógica pasando los parámetros necesarios desde la pantalla
- Para las colecciones no use arreglos []