UNIVERSIDAD SAN CARLOS DE GUATEMALA
FACULTAD DE INGENIERÍA
Organización Lenguajes y Compiladores 1
Escuela de Ciencias y Sistemas
Segundo Semestre 2018
Ing. Mario Bautista, Manuel Castillo, César Batz
Tutores Académicos
Luis Paz, José Soc, Jorge Castañeda

PROYECTO 1: Stores Generator

Objetivos

Generales:

- Que el estudiante sea capaz de implementar en una aplicación la fase de análisis léxico y sintáctico del compilador explicados en clase y laboratorio.

Específicos:

- Que el estudiante sea capaz de implementar acciones en un compilador, aplicándolos a temas útiles en la vida real.
- Que el estudiante se familiarice con el paradigma cliente-servidor.
- Que el estudiante utilice herramientas que están de punta en el mercado actual.

Descripción General del Problema:

En los últimos tiempos está muy de moda utilizar aplicaciones web para distintos tipos de problemas.

Se le llama SAAS a un servicio que se presta en la web, para simplificar procesos y optimizarlos a cambio de algún costo. La empresa Bussines Intelligence (BI) quiere contratarlo a usted como programador para llevar a cabo un proyecto innovador llamado Store Generator.

Store Generator es un SAAS que permite al cliente publicar su tienda incluyendo productos y demás a través del portal oficial de StoreGenerator.

Muchos usuarios dejan pasar la oportunidad de dar a conocer su mercancía a través de la web, esto es una gran limitante para ellos ya que limitan su publicidad solo sobre su entorno, y esto les quita la oportunidad de crecer. Aprovechándose de este fenómeno BI ha tomado la iniciativa y quiere proveerles a los usuarios una aplicación web (SAAS) que sea capaz de producir publicidad web para determinado negocio.

La finalidad de Store Generator es brindar un servicio que sea fácil de usar para el usuario, utilizando un lenguaje adaptable, flexible y legible (Store Lenguage), para que todo aquel usuario que no sepa que hacer para adentrarse al mundo de la publicidad web, pueda entrar y fácilmente publicar todo acerca de su negocio (Productos, Marcas, Precios, Ofertas, etc.).

Así mismo BI le ha pedido a usted que el servidor sea de alta demanda y sea capaz de soportar mucha cantidad de información y muchos usuarios conectados al mismo tiempo.

El servidor tendrá una terminal donde se puedan aplicar operaciones, determinadas por el lenguaje Store Lenguage. Toda operación hecha en el SAAS debe pasar por el servidor a través de dicho lenguaje.

Toda operación solicitada por el cliente debe ser pedida al servidor, de no haber errores de cualquier tipo, el servidor procederá a realizarla y responderle al cliente.

Arquitectura de la Aplicación

Servidor

El servidor es el eje central la aplicación, es el encargado de administrar toda la información, de usuarios, de tienda, de productos, etc. También gestionará la página web donde los clientes de la aplicación (usuario final) podrán ver toda la información de nuestras tiendas registradas.

La forma en la que el servidor se va a comunicar tanto con el almacenamiento como con el Usuario es por medio de Store Lenguage el cual se irá explicando conforme se vayan viendo los diferentes componentes de la aplicación.

Es de vital importancia mencionar que no hay otra forma en la que el servidor se comunique con los demás componentes.

Usuario del SAAS

Log In

Cuando un usuario del SAAS deseé ingresar al sistema, deberá identificarse como miembro del mismo, la solicitud que se enviará al servidor será la siguiente:

```
$request$
$InicioUsuario$
        $id$ 1 $id-$
        $password$ a1234b $password-$
$InicioUsuario-$
$request-$
```

$request$ Se usa cada vez que se quiere hacer una petición al servidor

Respuesta del Servidor

```
$reply$
$Usuario$
        $id$ 1 $id-$
        $access$ True $access-$
$Usuario-$
$reply-$
```

$reply$ Se usa cada vez que el servidor responde a las peticiones que se le hacen.

$usuario$ Se usa cada vez que hacen operaciones sobre el usuario.

$access$ Retorna, **True** si el password y el id del usuario coincidieron, **Falso** en caso de que no coincidan los datos y **Fail** si el usuario no existe.

Registrarse

La información que el SAAS necesita del usuario es la siguiente: ID, Nombre, Apellido, Password, teléfono, email, dirección (estas últimas dos opcionales).

Cuando se deseé registrar a un usuario, el usuario enviara la siguiente solicitud al servidor:

```
$request$
$CrearUsuario$
        $id$ 1 $id-$
        $nombre$ "nombre del usuario" $nombre-$
        $apellido$ "apellido del usuario" $apellido-$
        $password$  a1234b $password-$
        $telefono$ 12345678 $telefono-$
        $email$ correo@compiladores1.com $email-$
        $direccion$ "dirección del usuario" $dirección-$
$CrearUsuario-$
$request-$
```

$password$ Se debe validar que tenga más de 5 caracteres y que empiece y termine con una letra.

id El número que va dentro de esta etiqueta, debe ser único, en caso de no serlo, el servidor debe notificarlo como un **error.**

$email$ Esta etiqueta solo recibe correos **@compiladores1.com**

$dirección$, $teléfono$ estas etiquetas pueden venir o no venir, ya que se dice que no es una información vital para no registrar a un cliente, y que se puede hacer más adelante.

Respuesta del Servidor

```
$reply$
$Usuario$
        $registro id = 10$ True $registro-$
$Usuario-$
$reply-$
```

$registro$ la etiqueta viene acompañada del id del usuario que se está tratando y de los valores que puede tomar esta etiqueta son únicamente True, False dependiendo si se pudo o no realizar el registro.

Tiendas y Productos

Tienda

Las tiendas son la parte fundamental del SAAS ya que el motivo de este es promocionarlas tanto a ellas como a sus productos, para que los Usuarios tengan la oportunidad de expandir la frontera de su negocio. A continuación se hace énfasis en algunas características de la misma:

- Un usuario del SAAS puede tener muchas tiendas registradas.
- Una tienda puede tener muchos productos registrados
- Una tienda NO puede pertenecer a más de un Usuario del SAAS.

Registro de Tienda

Cuando se registra una tienda se requiere la siguiente información: código, id del propietario (usuario del SAAS), nombre de la tienda, dirección de la tienda, teléfono de la tienda.

```
$request$
        $tienda tipo= "crear"$
            $codigo$ 8 $codigo-$
            $propietario$ 5 $propietario-$
            $nombre$ "Librería Tatty" $nombre-$
            $direccion$ "Calle Real" $direccion-$
            $telefono$ 12345 $telefono-$
            $img$ "path de la imagen" $img-$
        $tienda-$
$request-$
```

img esta etiqueta hace referencia a la imagen que tendrá el objeto en la página web, debe ser extraída de la pc donde se ejecute el programa Cliente y llevada hacia el servidor.

El atributo **tipo** debe de venir dentro de la etiqueta tienda y especifica la operación que se desea realizar sobre latienda.

Respuesta del Servidor

```
$reply$
$tienda$
        $registro id = 5$ True $registro-$
$tienda-$
$reply-$
```

Modificar Atributos de una tienda.

La modalidad del lenguaje para modificar atributos de una tienda es de “una línea”, lo que significa que en una sola línea vendrán todos los cambios que se quieran realizar sobre la tienda, A continuación, se describen las características más importantes:

- Los atributos pueden venir en cualquier orden
- En la línea solo aparecen los atributos que se desean modificar
- Cualquier atributo que tenga la tienda se sobrescribirá, menos el código de la tienda, si alguien desea modificar este, el servidor devolverá un error.
- El atributo tipo, código y propietario siempre deben venir, el lenguaje asumirá que estos son los parámetros para referirse a la tienda y que estos nunca podrán ser cambiados.
- Se debe validar que la tienda pertenezca al usuario que está haciendo la petición, en caso de no serlo, se devolverá un error.

```
$request$
    $tienda tipo=”modificación”,
    código=100,
    Propietario=2,
    nombre= ”Librería Tatiana” ,
    dirección=”Guatemala”,
    teléfono=12345678 -$ //notar que toda es la misma línea
        //puede venir n cantidad de veces
        $tienda código=100, propietario=2, tipo=”modificación”, dirección=”Puerta Parada”-$
        $tienda código=100, propietario=2, teléfono=2222, tipo=”modificación”, -$
$request-$
```

NOTA: la etiqueta img no se puede modificar.

Respuesta del Servidor

```
$reply$
$tienda$
        $modificar id = 10$ True $modificar-$
$tienda-$
$reply-$
```

$modificar$ Esta etiqueta viene acompañada del id del objeto que se está tratando, en este caso tienda, y sus valores únicamente pueden ser True o False, si se pudo modificar o no.

NOTA: En caso de que se hayan modificado varias tiendas o la misma tienda en varias líneas como esta en el ejemplo el servidor debe de enviarle una respuesta por cada vez que se hizo modificación.

Eliminar una tienda.

Eliminar una tienda también usa la misma modalidad que el modificar una tienda a diferencia que este solo puede traer los atributos código y propietario para identificar que tienda es la que se desea eliminar y por último la palabra reservada tipo especificando que se desea la opción de **"eliminar",** es importante mencionar que también se deben verificar que la tienda pertenezca al usuario antes de eliminarla.

```
$request$
    $tienda código=100,
    Propietario=2,
    Tipo="Eliminar" –$
    // puede venir varias veces dentro de la etiqueta request
    $tienda código=3, propietario=2, tipo="eliminar" -$
$request-$
```

Respuesta del Servidor

```
$reply$
$tienda$
        $eliminar id = 10$ True $eliminar-$
$tienda-$
$reply-$
```

$eliminar$ Esta etiqueta viene acompañada del id del objeto que se está tratando, en este caso tienda, y sus valores únicamente pueden ser True o False, si se pudo eliminar o no.

NOTA: En caso de que se hayan eliminado varias tiendas como esta en el ejemplo el servidor debe de enviarle una respuesta por cada vez que se eliminó.

Producto

Como todos sabemos el sentido fundamental del SAAS es que el usuario tenga facilidad de promocionar sus productos, y sobre todo que el usuario pueda conocerlos a detalle, es por eso que se requiere una buena administración de estos, las características principales son las siguientes:

- Una tienda puede tener varios productos.
- Un producto solo puede pertenecer a una tienda.
- Dado a que un producto puede estar en varias tiendas, es importante mencionar que para no complicar la administración de la aplicación, si este caso se da, el producto deberá de registrarse el número de veces que aparezca en **cada tienda** como si fueran productos distintos, de esta forma seguimos cumpliendo con la segunda regla**.**
- Dos productos pueden tener el mismo código, pero en diferente tienda.

Espacio Operacional Aritmético (EOA)

Muchas veces describir los precios de un producto, o definir las cantidades es bastante difícil ya que debemos de tener a la mano una calculadora, para hacer las operaciones correspondientes y calcularlo, para solventar ese problema el lenguaje admite espacio operacional en la parte de productos el cual se describe de la siguiente manera:

- El espacio operacional aritmético puede ir dentro de cualquier etiqueta del producto.
- Si la etiqueta es nombre y usamos espacio operacional allí, el servidor nos debe de lanzar un error, ya que no tendría sentido tener un producto con nombre 789012.
- Para indicar un espacio operacional aritmético es necesario que vaya dentro de llaves
- El único símbolo de agrupación reconocido por EOA es **(**

Ejemplo

{(7*8+17*1)/(10+10)**}**

El resultado debería de ser 3.65

Las operaciones que soporta el EOA son las siguientes:

Operación	Operador
Suma	Expresión + Expresión
Resta	Expresión – Expresión
Multiplicacion	Expresión * Expresión
Division	Expresión / Expresión
Raiz Cuadrada	Rq(Expresión)
Potencia	Ptn(Expresión)

Registro de Producto

El sistema necesita registrar de un producto su nombre, la cantidad que se tiene en stock del producto, la marca del producto, el color del producto (se manejará un solo color por producto) y el tamaño del producto en cm.

```
$request$
$producto tipo="crear"$
        //solo las etiquetas que aparezcan se van a modificar
        $código$ 2 $código-$
        $nombre$ "Tasa" $nombre-$
        $cantidad$ {8*33+1} $cantidad-$//Es lo mismo que $cantidad$ 265 $cantidad-$
        $marca$ "Montoya" $marca-$
        $color$ "blanco" $color-$
        $tamaño$ {Pt(Rq((9+10)+10))} $tamaño-$
        $img$ "path de la imagen" $img-$
        $sucursal$ 3 $sucursal-$
$producto-$
$request-$
```

$sucursal$ esta etiqueta contendrá el código de la tienda en donde se distribuye dicho producto.

Respuesta del Servidor

```
$reply$
$producto$
        $registro id=10 sucursal=11$ True $registro-$
$producto-$
$reply-$
```

Modificar un Producto

La metodología para modificar un producto es “en línea”, esencialmente son las mismas reglas que en el caso de modificar una tienda, la única diferencia es que ahora las etiquetas que siempre deben de venir son tipo, código, sucursal. Es importante mencionar que dentro de estas puede venir un EOA.

```
$request$
$producto tipo=”modificar” codigo=10 sucursal=12 nombre=”Plato” cantidad= {8*8} -$
//Pueden venir muchas modificaciones dentro del mismo request e incluso eliminaciones
$producto tipo=”modificar” codigo=10 sucursal=12 nombre=”Plato” cantidad={8*8-1}-$
$request-$
```

Si la etiqueta de producto viene dentro de una etiqueta de tienda ya no es necesario identificar la sucursal

```
$request$
$tienda id=10$
$producto tipo=”modificar” codigo=10 nombre=”Plato” cantidad= {8*8} -$
//Pueden venir muchas modificaciones dentro del mismo request e incluso eliminaciones
$producto tipo=”modificar” codigo=10 nombre=”Plato” cantidad= {8*8 -1} -$
$tienda-$
$request-$
```

Respuesta del Servidor

```
$reply$
$producto$
        $modificar id=10 sucursal=11$ True $modificar-$
$producto-$
$reply-$
```

NOTA: En caso de que se hayan modificado varios productos o el mismo producto como esta en el ejemplo, el servidor debe de enviarle una respuesta por cada vez que se hizo modificación.

Eliminar un Producto

Al igual que el modificar, esta operación también es "en línea" y puede venir muchas veces dentro del mismo request. Los únicos parámetros que deben de venir son tipo, código, sucursal.

```
$request$
$producto  tipo="eliminar"  codigo=10 sucursal=12-$
$producto  tipo="eliminar"  codigo=11 sucursal=12-$
$producto  tipo="eliminar"  codigo=10 sucursal=13-$
$request-$
```

```
$reply$
$producto$
        $eliminar id=10 sucursal=11$ True $eliminar-$
$producto-$
$reply-$
```

Respuesta del Servidor

Almacenamiento

El almacenamiento es una parte muy importante en el sistema, ya que si no lo manejamos de forma correcta, el sistema no sería estable, se requiere que el sistema sea rápido ante las consultas de los datos y que los datos persistan en el tiempo, es decir, se puede caer el servidor y después levantarse y los datos deben de resguardarse de forma íntegra.

Tablas de Almacenamiento

Una tabla es la representación de los datos en el almacenamiento, es decir, una tabla está conformada por registros(tuplas) en su eje "y" y por atributos en su eje "x", una de las formas más eficiente de guardar datos masivos es la tabla hash, que precisamente es la estructura de datos más rápida para búsquedas lineales debido a su búsqueda por indexación, BI le recomienda usarla, mas sin embargo no es indispensable el uso de la misma, podemos usar otro tipo de Estructura de datos, lo que es indispensable, es que esta misma sea almacenada en el disco duro por medio de archivos.

Las tablas indispensables para el sistema son las siguientes:

- Usuarios
- Tiendas
- Productos

Usted como ingeniero en sistemas deberá de pensar como darle solución a los requerimientos de la empresa.

Valor Vacío

Este valor se utilizará en el almacenamiento, para aquellos valores que no son indispensables y que no tienen un valor especificado, haciendo referencia a que no existe valor para ese campo de determinada tupla.

Consultas sobre el Almacenamiento

Las consultas sobre el almacenamiento solo pueden ser realizadas por el servidor y estas se hacen mediante la etiqueta **$query$,** como todos sabemos una consulta es extraer datos del almacenamiento.

Operadores Lógicos (OL)

Los operadores lógicos soportados por el sistema son los siguientes.

Operación	Operador
AND	%%
OR	##
NOT	N(Expresión)

Extracción de datos.

Por ser la primera versión del sistema se van a manejar extracciones de datos bastante básicas, las extracciones se hacen mediante el operador **extraer** y su sintaxis es la siguiente

Extraer de [nombre de la tabla];

Extraer de [nombre de la tabla] donde [Expresión Lógica];

La expresión lógica es una cláusula de comparaciones donde se utilizarán los Operadores Lógicos si se desea, por ser la primera versión el único comparador que se utilizara será el "==", que lo único que hace es verificar que ambos lados sean iguales. Si se desea utilizar una expresión lógica la palabra reservada para indicarlo es **donde.**

En [nombre de la tabla] solo pueden venir las tres tablas de nuestro sistema, las cuales se describieron anteriormente (Usuario, Tienda, Producto)

Ejemplos de Expresiones Lógicas.

Tamaño==12 ## Tamaño==11

Nombre=="Tienda de Don Justo"

Nombre=="Tasa" %% (Precio==10 ## Precio==2)

NOTAS IMPORTANTES:

- Siempre se devuelven todos los atributos de las tuplas que devuelva la consulta.
- Si se elige la tabla Usuario, por ejemplo, en las expresiones lógicas solo se podrán comparar atributos de la tabla usuario y de ninguna otra, para que el sistema no se vuelva demasiado complejo.

Ejemplo 1 de Consultas

```
$query$
        Extraer de Usuario;
$query-$
```

Ejemplo 2 de Consultas

```
$query$
        Extraer de Usuario donde Nombre==”Justo” %% N(Telefono==Vacio);
$query-$
```

Ejemplo 3 de Consultas

```
$query$
        Extraer de Producto donde Cantidad=={8*8} ## Precio==20;
$query-$
```

NOTAS IMPORTANTES:

- Las etiquetas request y query no son compatibles, es decir, no pueden venir juntas en la misma compilación.

Manejo de Errores

La aplicación deberá crear reportes de los errores Léxicos, Sintácticos y Semánticos que se encuentren durante la compilación. Los errores se pueden detectar ya en el ingreso manual de request al servidor o en el ingreso de búsquedas, por lo que se deben considerar ambos.
Dicho reporte de errores debe mostrar de forma clara, ordenada y categorizados, los errores, es decir, deben mostrar primero los errores léxicos, luego los sintácticos y por último los semánticos. Además, deberá mostrar la posición donde ocurrió el error (fila-columna) y qué provocó el error. Se evaluará la capacidad de la aplicación de recuperarse de los errores, esto quiere decir, si se suspende la ejecución informando al usuario el error encontrado para que lo pueda corregir y volver a compilar, o si la compilación continúa su ejecución de forma transparente al usuario hasta donde sea capaz para luego mostrar el reporte de los errores encontrados.

Reportes de Errores

Todos los Reportes de los errores descritos anteriormente deberán presentarse en páginas realizadas formato PDF.

Respuesta del Servidor para notificarle algún error al Usuario

```
$reply$
$error$
        $descripcion$ "El path que ingreso de imagen no sirve" $descripcion-$
$error -$
$reply-$
```

Aplicación del Servidor

Consola

Esta será un campo de texto que recibirá Store Lenguage para su compilación y posterior ejecución.

Ejecutar Acciones de la Consola

Este es el botón que servirá para dar inicio a la compilación del Store Lenguage que se encuentre en la consola. Los resultados se verán en la consola de salida.

Archivo de Entrada

Muchas veces es dificultoso hacer cargas de datos, manualmente, ya que tenemos que ingresar dato por dato, por eso el Servidor tiene la modalidad de Archivo de Entrada, donde se ejecutará un script con “Store Lenguage” y en caso de no tener errores, se deberá generar todas las peticiones al servidor. Dichos archivos tendrán extensión .SL

Consola de Salida

El Servidor debe de tener una consola donde aparezca el resultado de una consulta. Esta consola no debe ser modificable, solo para salida de datos.

La consola es la encargada de notificar todas las operaciones realizadas por el usuario, por ejemplo si se crea un usuario la consola tiene que mostrar un mensaje reportando que fue ingresado al almacenamiento, también se deberán de reportar errores, cuando un solo script lleva varias instrucciones, por ejemplo dos modificaciones sobre varios productos, estos mismos se deberán de ver reflejados en la consola, cada uno en una nueva línea.

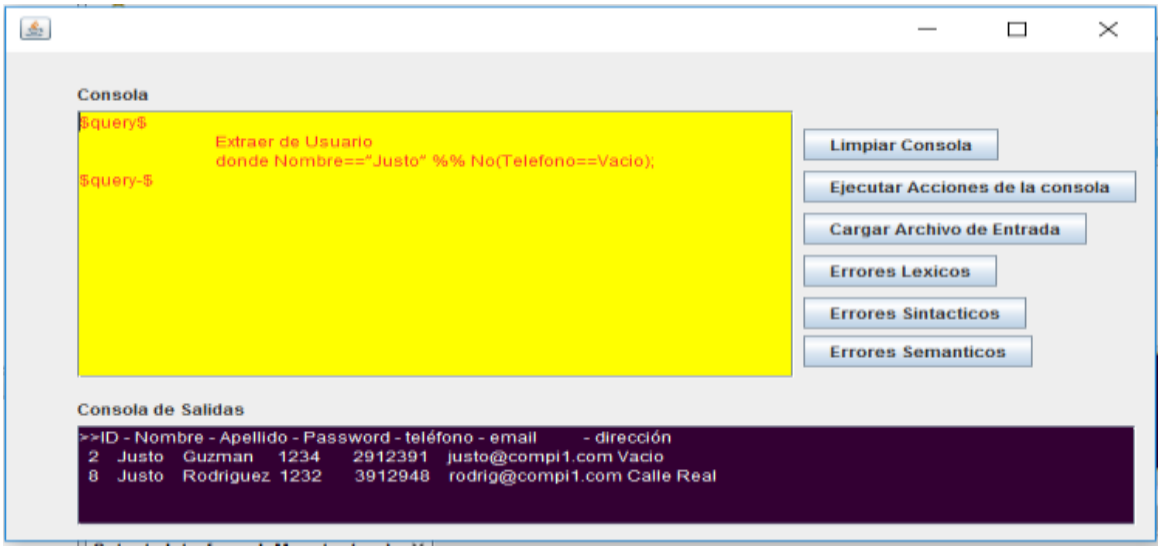
Por cada salida la consola debe de anteponer el símbolo >> por ejemplo:

```
>> Se creó el Usuario Juan exitosamente con id 1

>> No se puedo crear el usuario Jorge porque el ID ya está en uso
```

Limpiar Consola

Al momento de presionar este botón la consola quedará vacía para un nuevo script y la consola de salida también se deberá limpiar.



NOTA: Se verificará que la interfaz sea muy amigable para el usuario.

Aplicación para Usuario del SAAS

Interfaz Grafica

Log In

La interfaz gráfica deberá presentar una pantalla para que el usuario del SAAS pueda loguearse.

Los pasos para realizar esta acción son:

1. El usuario ingresa su nombre y contraseña.
2. El usuario presiona el botón de “Entrar”
3. La aplicación del Usuario convierte esa entrada a Store Lenguage descrito anteriormente para enviarlo al servidor.
4. El Servidor responde a la petición, la aplicación del usuario compila la respuesta del servidor y determina si dejar pasar al usuario o no.

Sección de Tiendas

Cuando el usuario ingrese al sistema, podrá ver todas las tiendas que tiene registradas y navegar entre ellas, podrá entrar a ver sus productos, modificarlas desde la interfaz o eliminarlas, los pasos para esto son los siguientes:

Si el usuario elige la opción “entrar” se pasará a otra pantalla donde van a aparecer todos los productos para la tienda que se eligió.

Si el usuario elige modificar:

1. Aparecerá un menú donde van a salir los atributos para la tienda donde se ingresarán los atributos que se deseen cambiar, los que se dejen en blanco, se asumirá que se desea que se queden como están.
2. Se convertirá todo en código Store Lenguage y se enviará al servidor.
3. El servidor responde.

Si el usuario elige eliminar:

1. Se convertirá todo en código Store Lenguage y se enviará al servidor.
2. El servidor responde.

Petición al Servidor para devolver lista de Tiendas

Cuando el usuario entre al sistema, este mismo deberá de enviar una petición al servidor para que le devuelva la lista de sus tiendas, esto se hará de la siguiente manera:

```
$request$
$get tipo="tiendas" propietario=2 -$
$resquest-$
```

Respuesta del Servidor para devolver la lista de tiendas

```
$reply$
$lista$
        $tienda$
                $codigo$ 8 $codigo-$
                $propietario$ 5 $propietario-$
                $nombre$ "Librería Tatty" $nombre-$
                $direccion$ "Calle Real" $direccion-$
                $telefono$ 12345 $telefono-$
        $tienda -$
        $tienda$
                $codigo$ 10 $codigo-$
```

$propietario$ 5 $propietario-$

$nombre$ ”Discoteca Movil” $nombre-$

$direccion$ ”Calle del Arco” $direccion-$

$telefono$ 12343 $telefono-$

$tienda -$

$lista -$

$reply-$

NOTA: Pueden venir cualquier cantidad de tiendas dentro de la lista.

Sección de Producto

Cuando el usuario presione la opción de entrar a alguna tienda deberá aparecer un menú con todos los productos que contiene esa tienda en específico, también se deberán de mostrar las opciones de modificar y eliminar y se deberá de realizar los mismos pasos que en el modificar y eliminar de lastiendas.

La forma de pedir la lista de productos es análoga a la lista de tiendas, a continuación, un ejemplo: El cliente enviará la siguiente petición:

$request$

$get tipo=”productos” propietario=2 sucursal=3 -$

$resquest-$

Respuesta del Servidor para devolver la lista de productos

$reply$

$lista$

$producto$

$código$ 2 $código-$

$nombre$ ”Tasa” $nombre-$

$cantidad$ 8 $cantidad-$

$marca$ ”Montoya” $marca-$

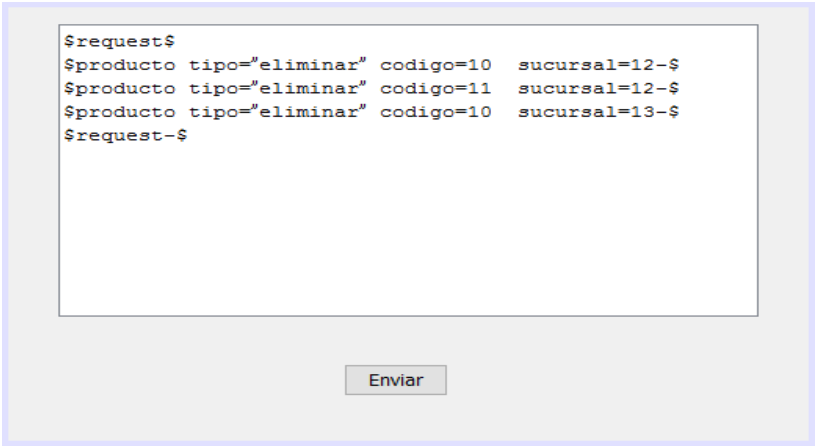
$color$ ”blanco” $color-$

```
        $tamaño$ 7.1 $tamaño-$
        $sucursal$ 3 $sucursal-$
$producto-$
$producto$
        $código$ 9 $código-$
        $nombre$ ”Consola” $nombre-$
        $cantidad$ 1 $cantidad-$
        $marca$ ”Xbox” $marca-$
        $color$  ”negro” $color-$
        $tamaño$ 20 $tamaño-$
        $sucursal$ 3 $sucursal-$
$producto-$
$lista-$
$reply-$
```

NOTA: Pueden venir cualquier cantidad de productos dentro de la lista.

Consola

La aplicación del usuario deberá de contar con una consola de datos donde se podrán hacer peticiones al servidor a través de Store Lenguage



Página Web

Esto es conocido como el FrontEnd de la aplicación, es decir es la capa del sistema con la que el usuario final va a interactuar, esta misma va a estar corriendo en un Servidor web y administrada por un lenguaje de programación web, la empresa BI le ha pedido a usted como programador tener estricto cuidado con el **estilo de la página web** ya que tiene que ser vistosa para que el usuario pueda navegar cómodamente.

Tiendas en la página web

La página web es sencilla únicamente nos mostrara un catálogo con todas las tiendas que tenemos registradas en nuestro sistema, es fundamental que cada tienda tenga su imagen con la que fue registrada, en caso de no tener imagen se debe mostrar una imagen por default. Así mismos se deben de mostrar todos los atributos de las tiendas a excepción del código.

Productos en la página web

Al darle clic a una tienda se podrán ver todos los productos que se tienen registrados en el sistema con su respectiva imagen y atributos a excepción de su código.

Documentación

- Manual Técnico
- Manual de Usuario

Restricciones

- El proyecto debe realizarse de forma individual.
- El proyecto debe ser implementado en el lenguaje JAVA, en caso contrario, este no será calificado y se tendrá una nota automática de 0.
- Para leer los archivos de entrada, se deben utilizar las herramientas JFlex y Cup.
- Se tomará en cuenta la calidad de la información proporcionada por el compilador cuando se produzcan errores, así como la presentación de la interfaz gráfica y amigabilidad de la aplicación.
- Copias de proyectos o de gramáticas tendrán una nota de 0 puntos y el respectivo reporte al Ingeniero, así como también el reporte a la Escuela de Ciencias y Sistemas.
- La calificación del proyecto será personal y durará como máximo 40 minutos, en un horario que posteriormente será establecido. Se debe tomar en cuenta que durante la calificación no podrán estar terceras personas alrededor o de lo contrario no se calificará el proyecto.
- Todos los errores generados por la aplicación deberán ser mostrados en PDF o de lo contrario no serán calificados.
- No se permitirá el uso de ninguna librería para Parsear XML.
- Sistema Operativo Libre
- Para administrador de página web debe ser JavaScript (Podrán utilizar algún framework como Angular 4, React, etc. si lo desean)
- El servidor para levantar la página web debe ser Node Js
- La entrega será en por classroom
  - Aplicaciones Java, El nombre debe tener el siguiente formato:
  - ProyectoUno_#carnet.JAR
  - Manuales de Usuario y Técnico
  - Código Fuente

Fecha de entrega y calificación

- La fecha de entrega está programada para el día XXXX de XXXX del 2018.
- La fecha y el lugar de la calificación se darán a conocer en días próximos.