Objetivos generales

- Aplicar conocimientos de analisis lexico y sintactico, así como el manejo de errores para crear herramientas útiles.

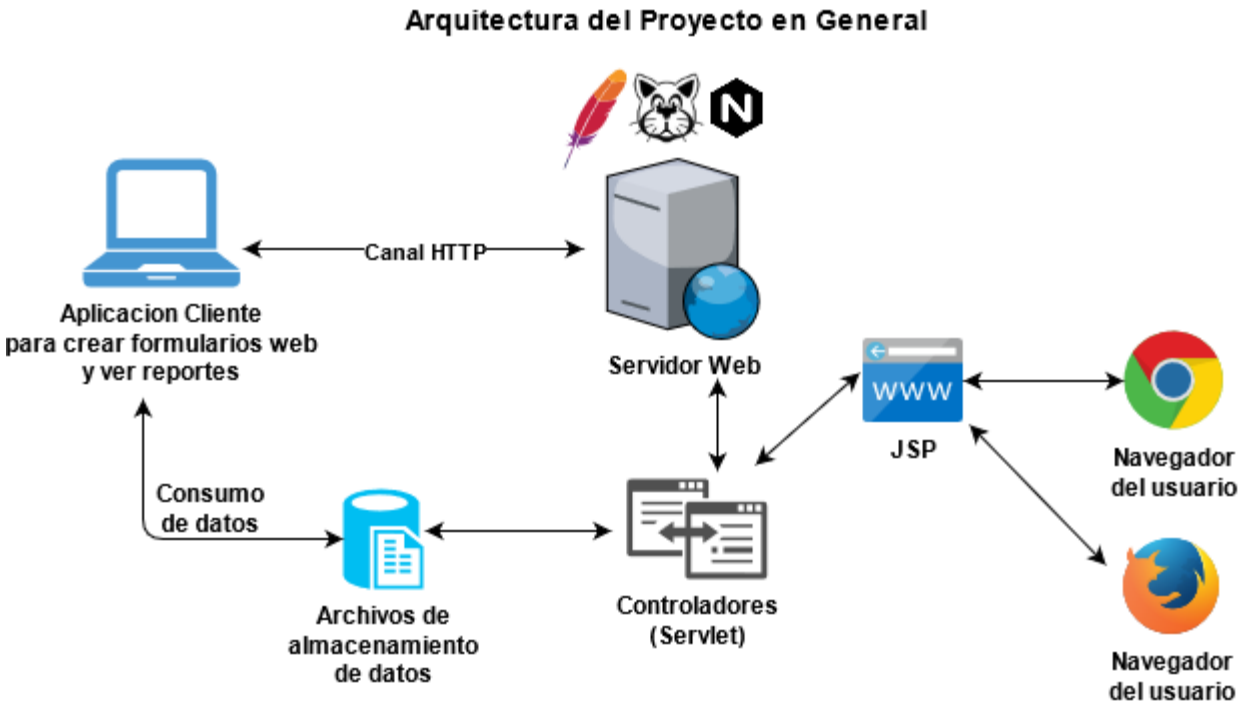
Objectivos específicos

- Crear expresiones regulares y gramaticas libres de contexto complejas.
- implementar las fases de análisis léxico y sintáctico de un compilador.
- Combinar la funcionalidad de JFlex y Cup en aplicaciones reales.
- Entender una arquitectura Cliente-servidor.
- Adquirir conocimientos sobre los lenguajes XML y JSON.

Descripción de la actividad

Hoy en día con la nueva dinámica de hacer todo de forma remota y digital, surgen algunas necesidades bastante interesantes en el ámbito académico, y es que existen ocasiones en donde un docente necesita recabar información de distintas índoles por medio de formularios como las encuestas, o incluso hacer cuestionarios con preguntas para los estudiantes que tiene a su cargo. Pues si bien es cierto ya existen muchas herramientas que cumplen con este propósito, el problema radica en que muchas veces les resulta un poco complejo a la mayoría de usuarios encontrar una herramienta adecuada y fácil de usar para poder generar formularios de forma rápida, ágil y sin mayores complicaciones.

Tomando el contexto anterior, se le contrata a usted como especialista en soluciones de TI para la creación de un sistema que permita la creación de formularios web a través de instrucciones con una estructura similar a los formatos XML y JSON, la idea es poder brindar al usuario del sistema una forma más a la medida para que puedan darse el lujo de crear varios tipos de modelos de formularios que le sirvan para hacer encuestas, cuestionarios, o cualquier otro modelo que tenga implícito la creación de un formulario. La arquitectura que se pretende adaptar es la siguiente.



Funciones del servidor

Las páginas web que contienen los formularios creados por el usuario deben ser servidas por medio de un servidor web HTTP (Tomcat, Apache, Nginx, JBoss, o similar para aplicaciones web java) a través del puerto 80.

Los usuarios finales que tendrán acceso a los formularios deben conectarse al servidor por medio de navegadores web y así navegar hacia la página que contiene un formulario en particular. **Un formulario debe ser creado por un usuario del sistema** previamente registrado, la idea es que un usuario del sistema debe tener la **posibilidad de autogenerar links de los formularios** que ha creado y así mismo poder **compartirlos a los usuarios finales** o público en general que tenga acceso al servidor por medio de su navegador. Cada usuario del sistema debe tener la posibilidad de loguearse tanto en la aplicación cliente, como en el sitio principal del servidor en donde podrá visualizar (en forma de lista, tabla, etc) los formularios disponibles que haya podido crear en su momento y así mismo generar los links de acceso que se discutirán más adelante.

En el servidor también se ejecutan los controladores (Servlets) que deben permitir la comunicación entre la aplicación cliente y el servidor web, para poder manejar todo tipo de solicitud y **mensajes estructurados en un lenguaje llamado Indigo** que es un código híbrido entre XML y JSON de la siguiente forma:

Una sola solicitud:

```
<!ini_solicitud:"NOMBRE_DE_LA_SOLICITUD">
      { "bloque_parametros":[{
            "parametro_1_serie1": "valor_parametro_1_serie1",
            "parametro_2_serie1": "valor_parametro_2_serie1",
            "parametro_N_serie1": "valor_parametro_N_serie1",
          },
          {"parametro_1_serie2": "valor_parametro_1_serie2",
           "parametro_2_serie2": "valor_parametro_1_serie2",
           "parametro_N_serie2": "valor_parametro_N_serie2",
          },
          { "parametro_1_serieM": "valor_parametro_1_serieM",
            "parametro_2_serieM": "valor_parametro_2_serieM",
            "parametro_N_serieM": "valor_parametro_N_serieM",
          }
        ]
      }
<fin_solicitud!>
```

Más de una solicitud:

```
<!ini_solicitudes>
      <!ini_solicitud:"NOMBRE_DE_LA_SOLICITUD_1">
           { "bloque_parametros":[{
                ....
                },
                ....
              ]
           }
      <fin_solicitud!>
       <!ini_solicitud:"NOMBRE_DE_LA_SOLICITUD_2">
           { "bloque_parametros":[{
                ....
                },
                ....
              ]
           }
      <fin_solicitud!>
      ....
      ....
<!fin_solicitudes>
```

Las respuestas del Servidor deben seguir el mismo lenguaje de estructuración Indigo con las variantes siguientes:

```
<!ini_solicitudes> se debe cambiar por <!ini_respuestas>

<!fin_solicitudes> se debe cambiar por <!fin_respuestas>

<!ini_solicitud:"NOMBRE_DE_LA_SOLICITUD">  se cambia por
<!ini_respuesta:"NOMBRE_DE_LA_RESPUESTA">

<fin_solicitud!> se debe cambiar por <!fin_respuesta>
```

Los parámetros que se deseen enviar como r**espuesta al cliente, quedan a criterio propio** ya que va a depender del nivel de detalle que desee manejar, el cual es muy importante también **para el tratamiento de errores.**

Aplicación cliente

La aplicación cliente debe ser una **aplicación de escritorio** usada por los usuarios registrados para **enviar solicitudes y recibir respuestas del servidor.** Para poder habilitar la comunicación al servidor se debe contar con un usuario y una contraseña, que se deben enviar en forma de solicitud al servidor para que este mismo les responda si pueden o no iniciar la comunicación. Los detalles de creación de usuarios se describen más adelante.

La aplicación cliente debe estar **compuesta por un área de texto donde el usuario escribe el código Indigo** de la o las solicitudes (las cuales se describen más adelante también), **un área para cargar (abrir) y guardar archivos de texto plano** con la posibilidad de mecanismos básicos de edición de texto (copiar, cortar, pegar, etc) **y un área de reportes** donde se desplieguen los datos registrados de cada formulario en forma de tabla. Para obtener estos reportes también se utilizara un **lenguaje de consultas llamado SQForm**, el cual se describe en el apartado de reportes.

Además de lo ya mencionado también se debe **contar con un área de respuesta** donde se visualizan:

- **Errores léxicos o sintácticos**: que se generan al ejecutar las solicitudes en código Indigo. Se debe incluir una descripción completa y clara del error (linea, columna, token o símbolo de error)
- **Mensajes de respuesta del servidor:** el servidor responde al cliente por medio de mensajes en código Indigo que deben ser entendidos por la aplicación cliente y desplegados en el área de respuesta.

Sitio de formularios web

El sistema debe ser capaz de contar con un **sitio general** que pueda manejar varias **páginas con formularios web** según el cliente que esté logueado. El servidor HTTP es el encargado de despachar las páginas web al puerto 80 pero los controladores deben ser los encargados de manipular cada formulario y **almacenar (ESTILO BASE DE DATOS) la información** que otros usuarios (Externos, con o sin una cuenta en el sistema) ingresen a través de los **formularios creados por un usuario del sistema**, esto quiere decir que las **páginas que contienen los formularios deben estar abiertas al público**. La información a almacenar serán los datos asociados a un formularios desplegado en el sitio (solo se permite un formulario dentro de cada página web creada). La estructura de almacenamiento de información se especifica al final.

Cabe recalcar que **para que un usuario pueda generar el link de acceso a sus formularios debe poder ingresar antes al sistema por medio de un login**. Una vez logueado el usuario debe poder ver todos los formularios que él pudo crear, y tener un mecanismo que le permita generar el link de acceso a un formulario en específico (el link debe funcionar solo de manera local).

Solicitud para crear un usuario del sistema

La aplicación cliente debe permitir enviar la solicitud para creación de un nuevo usuario del sistema. **Este mismo usuario servirá tanto para el login de la aplicación cliente como para el del sitio de formularios**.

Nombre de la solicitud: **CREAR_USUARIO**
Parámetros:

- **USUARIO*:** Nombre de usuario a registrar en el sistema (debe ser único y sin espacios).
- **PASSWORD***: Contraseña del usuario del sistema sin espacios.
- **FECHA_CREACION***: Fecha en que se creó el nuevo usuario, en formato yyyy-MM-dd

Si no se incluye explícitamente el parámetro **FECHA_CREACION,** la aplicación cliente debe agregarlo al mensaje con los valores correctos.

Ejemplo:

```
<!ini_solicitud:"CREAR_USUARIO">
      { "CREDENCIALES_USUARIO":[{
            "USUARIO": "juanito619",
            "PASSWORD": "12345678"
          }
        ]
      }
<fin_solicitud!>
```

Solicitud para modificar un usuario del sistema

La aplicación cliente debe permitir enviar la solicitud para modificar un usuario existente del sistema.

Nombre de la solicitud: **MODIFICAR_USUARIO**
Parámetros:

- **USUARIO_ANTIGUO*:** Nombre de usuario que se desea cambiar.
- **USUARIO_NUEVO*:** Nombre del nuevo usuario (debe ser único).
- **NUEVO_PASSWORD***: Nueva contraseña del usuario del sistema.
- **FECHA_MODIFICACION***: Fecha de la última modificación.

Ejemplo:

```
<!ini_solicitud:"MODIFICAR_USUARIO">
      { "CREDENCIALES_USUARIO":[{
            "USUARIO_ANTIGUO": "juanito619",
            "USUARIO_NUEVO": "juanito619lopez",
            "NUEVO_PASSWORD": "12345678910"
          }
        ]
      }
<fin_solicitud!>
```

Si no se incluye explícitamente el parámetro **FECHA_MODIFICACION,** la aplicación cliente debe agregarlo al mensaje con los valores correctos.

Solicitud para eliminar usuarios del sistema

Al eliminarse un usuario del sistema, se deben eliminar también todos los formularios web que pudo crear en su momento.

Nombre de la solicitud: **ELIMINAR_USUARIO**
Parámetros:

- **USUARIO*:** Nombre del usuario a eliminar.

Ejemplo:

```
<!ini_solicitud:"ELIMINAR_USUARIO">
      { "CREDENCIALES_USUARIO":[{
            "USUARIO": "juanito619lopez"
          }
        ]
      }
<fin_solicitud!>
```

Solicitud para pedir acceso al sistema

Después de haberse creado un usuario se debe pedir acceso al sistema por medio del servidor, si este existe y las credenciales son correctas, el servidor le enviará una respuesta en el mismo formato Indigo, y tendrá que existir un mecanismo que lo entienda y despliegue la respuesta en el área correspondiente. **Si el servidor confirma la solicitud el usuario tendrá respuestas en las solicitudes que envíe**, y si

por otro caso no recibe confirmación, el servidor únicamente seguirá pidiendo la solicitud de login hasta que se de una correcta.

Nombre de la solicitud: **LOGIN_USUARIO**
Parámetros:

- **USUARIO*:** Nombre de usuario que desea ingresar al sistema).
- **PASSWORD*:** Contraseña del usuario.

Ejemplo:

```
<!ini_solicitud:"LOGIN_USUARIO">
     { "CREDENCIALES_USUARIO":[{
            "USUARIO": "juanito619",
            "PASSWORD": "12345678"
           }
         ]
      }
<fin_solicitud!>
```

Para desloguearse de la aplicación cliente, únicamente se debe habilitar un botón que después de haberse logueado el usuario, le permite poder cerrar sesión de forma simple.

Formularios web

Los u**suarios que están registrados en el sistema son los encargados de la creación de formularios web** usando componentes. Los componentes se agregan a cada página para construir el formulario y se pueden configurar de manera que el contenido tenga las propiedades deseadas por el usuarios.

Solicitud para crear formulario

Nombre de la solicitud: **NUEVO_FORMULARIO**
Parámetros:

- **ID*:** el identificador del formulario (debe ser único).
- **TITULO*:** El título del formulario mostrado en la página web, puede ser cualquiera.
- **NOMBRE*:** El nombre del formulario (no debe tener espacios).
- **TEMA***: Se debe contar con mínimo de dos temas que cambien la apariencia TOTAL del formulario, Ejemplo: Dark, Blue, White, entre otros.
- **USUARIO_CREACION***: el usuario que ejecutó la solicitud.
- **FECHA_CREACION***: Fecha en que se crea el formulario, en formato yyyy-MM-dd

Si no se incluye explícitamente alguno de los parámetros **USUARIO_CREACION, FECHA_CREACION,** la aplicación cliente los debe agregar al mensaje con los valores correctos.

Ejemplo:

```
<!ini_solicitud:"NUEVO_FORMULARIO">
      { "PARAMETROS_FORMULARIO":[{
            "ID": "$form1",
            "TITULO": "Formulario para encuesta 1",
            "NOMBRE": "formulario_encuesta_1",
            "TEMA": "Dark"
          }
        ]
      }
<fin_solicitud!>
```

Solicitud para eliminar formularios

Nombre de la solicitud: **ELIMINAR_FORMULARIO**
Parámetros:

- **ID*:** ID del formulario a eliminar.

Ejemplo:

```
<!ini_solicitud:"ELIMINAR_FORMULARIO">
      { "PARAMETROS_FORMULARIO":[{
            "ID": "$form1"
          }
        ]
      }
<fin_solicitud!>
```

Solicitud para modificar parámetros de formularios

Esta solicitud permite reemplazar los valores de los parámetros TITULO, NOMBRE y TEMA de un formulario ya existente. Además del parámetro **ID**, es obligatorio que alguno de los parámetros anteriores esté definido.

Nombre de solicitud: **MODIFICAR_FORMULARIO**
Ejemplo:

```
<!ini_solicitud:"MODIFICAR_FORMULARIO">
```

```
      { "PARAMETROS_FORMULARIO":[{
            "ID": "$form1",
            "TITULO": "Formulario Modificado para encuesta 1",
            "NOMBRE": "formulario_encuesta_1_v2",
            "TEMA": "Blue"
          }
        ]
      }
<fin_solicitud!>
```

Solicitud para agregar componentes a formulario web

Esta acción permite agregar un componente a un formulario específico.

Nombre de solicitud: **AGREGAR_COMPONENTE**
Parámetros:

- **ID*:** Identificador del componente. Este ID debe ser único en el contexto del formulario web.
- **NOMBRE_CAMPO*:** Que es el nombre con el cual se identifica el campo para el acceso a datos recopilados (no debe tener espacios).
- **FORMULARIO*:** Identificador del formulario en donde se va a crear el componente.
- **CLASE*:** la clase de componente. Puede ser CAMPO_TEXTO, AREA_TEXTO, CHECKBOX, RADIO, FICHERO, IMAGEN, COMBO, BOTON.
- **INDICE:** Representa el número de posición del componente dentro del formulario, este parámetro no debe especificarse al momento de agregar componentes ya que se debe **generar automáticamente**. Por ejemplo, si es el primer componente a agregar tendrá el índice 1, y así sucesivamente el próximo tendrá el 2, 3, 4...N. etc.
- **TEXTO_VISIBLE*:** Es la descripción visible del componente, como su título o la especificación de lo que requiere que se ingrese o envíe.
- **ALINEACION:** indica la alineación del componente junto a su descripción, el valor puede ser CENTRO, IZQUIERDA, DERECHA, JUSTIFICAR
- **REQUERIDO:** Marca si el componente es obligatorio de rellenar o seleccionar.
- **OPCIONES**: Solo cuenta en los componentes clase: CHECKBOX, RADIO, y COMBO. Además debe estar definido como: item1 | item2 | item3 | itemN…..
- **FILAS:** Solo aplica al componente AREA_TEXTO, y hace referencia al número de filas.
- **COLUMNAS:** Solo aplica al componente AREA_TEXTO, y hace referencia al número de filas.
- **URL**: Solo aplica para el componente IMAGEN, y hace referencia a la dirección url de la imagen a mostrar.
- En el caso de las **imágenes**, solo se debe especificar: ID, FORMULARIO, CLASE, URL, TEXTO_VISIBLE, ya que no se almacenará información que ingrese o cargue el usuario final en este componente.

- En el caso de los **botones**, únicamente se le debe especificar: ID, FORMULARIO, CLASE, TEXTO_VISIBLE, ya que al igual que las imágenes no almacenan información que el usuario final especifique o cargue. La programación del botón, ya depende de como se quiera manejar para procesar y guardar la información de los componentes de cada formulario.

Ejemplo 1:

```
<!ini_solicitud:"AGREGAR_COMPONENTE">
      { "PARAMETROS_COMPONENTE":[{
            "ID": "$_text_cliente",
            "NOMBRE_CAMPO": "Cliente",
            "FORMULARIO": "$form1",
            "CLASE": "CAMPO_TEXTO",
            "TEXTO_VISIBLE": "Nombre de cliente: ",
            "ALINEACION": "CENTRO",
            "REQUERIDO": "SI"
          }
        ]
      }
<fin_solicitud!>
```

Ejemplo 2:

```
<!ini_solicitud:"AGREGAR_COMPONENTE">
      { "PARAMETROS_COMPONENTE":[{
            "ID": "$_grupo_paises",
            "NOMBRE_CAMPO": "Pais",
            "FORMULARIO": "$form1",
            "CLASE": "COMBO",
            "TEXTO_VISIBLE": "Pais de Origen: ",
            "ALINEACION": "CENTRO",
            "REQUERIDO": "SI",
            "OPCIONES": "Guatemala|El Salvador|Honduras|OTRO"
          }
        ]
      }
<fin_solicitud!>
```

Solicitud para eliminar componentes de formularios web

Esta acción permite eliminar un componente a un formulario web específico.

Nombre de solicitud: **ELIMINAR_COMPONENTE**
Parámetros:

- **ID*:** Identificador del componente.
- **FORMULARIO*:** Identificador del formulario en donde se encuentra el componente a eliminar.

Ejemplo:

```
<!ini_solicitud:"ELIMINAR_COMPONENTE">
      { "PARAMETROS_COMPONENTE":[{
            "ID": "$_grupo_paises"
          }
        ]
      }
<fin_solicitud!>
```

Solicitud para modificar componentes de formulario web

Esta acción prácticamente reemplaza un componente existente en un formulario por la nueva configuración enviada en esta solicitud, si no se especifica algún parámetro que no es obligatorio el componente mantendrá la misma estructura del parámetro que se especificó al momento de crearlo. Sigue la misma estructura de la solicitud **AGREGAR_COMPONENTE**

Valor del atributo nombre: **MODIFICAR_COMPONENTE**
Parámetros:

- **ID*:** Identificador del componente. Este ID ya debe existir en el formulario web y se toma como referencia para hacer el cambio.
- **FORMULARIO*:** Identificador del formulario en donde se va a modificar el componente.
- **INDICE:** En este caso se debe especificar si se quiere cambiar la posición del componente dentro del formulario, por ejemplo si se especifica Índice = 1, el componente pasará a la primera posición del formulario.
- Todos los demás parámetros que se quieran modificar.

Ejemplo 1:

```
<!ini_solicitud:"MODIFICAR_COMPONENTE">
      { "PARAMETROS_COMPONENTE":[{
            "ID": "$_grupo_paises",
            "FORMULARIO": "$form1",
            "CLASE": "CHECKBOX",
            "INDICE": "1",
            "ALINEACION": "DERECHA",
```

```
            "OPCIONES": "Guatemala|El Salvador|Honduras|OTRO"
           }
         ]
      }
<fin_solicitud!>
```

Propuestas de estructuración de componentes solicitados

CAMPO_TEXTO

Ejemplo de etiqueta:

```
<label for="male">TEXTO_VISIBLE</label>
 <input type="text" id="id_etiqueta" name="nombre_campo" />
```

AREA_TEXTO

Ejemplo de etiqueta:

```
<label for="male">TEXTO_VISIBLE</label>
<textarea id="id_etiqueta" rows="numerofilas" cols="numerocolumnas"
name="nombre_campo"></textarea>
```

CHECKBOX

Ejemplo de etiqueta:

```
<label for="male">TEXTO_VISIBLE</label>
<input type="checkbox" id="id" name="nombre_campo" value="html">HTML
<input type="checkbox" id="id" name="nombre_campo" value="javascript">Javascript
<input type="checkbox" id="id" name="nombre_campo" value="css">CSS
<input type="checkbox" id="id" name="nombre_campo" value="xml">XML
```

RADIO

Ejemplo de etiqueta:

```
<label for="male">TEXTO_VISIBLE</label>
<input type="radio" id="id_etiqueta" value="menos18" name="nombre_campo"/>Menos de 18
<input type="radio" id="id_etiqueta" value="18a30" name="nombre_campo"/>18 a 30
<input type="radio" id="id_etiqueta" value="31a50" name="nombre_campo"/>31 a 50
<input type="radio" id="id_etiqueta" value="mas50" name="nombre_campo"/>Más de 50
```

FICHERO

Ejemplo de etiqueta:

```
<label for="male">TEXTO_VISIBLE</label>
 <input type="file" id="id_etiqueta" value="valor" name="nombre_campo"/>
```

IMAGEN

Ejemplo de etiqueta:

```
<label for="male">TEXTO_VISIBLE</label>
<input type="image" id="id_etiqueta" src="url-image" />
```

COMBO

Ejemplo de etiqueta:

```
<label for="male">TEXTO_VISIBLE</label>
<select id="identificador" name="nombre_campo" >
  <option>Atletico de Madrid</option>
  <option selected="selected">Betis</option>
  <option>FC. Barcelona</option>
  <option>Real Madrid</option>
  <option>Zaragoza</option>
</select>
```

BOTON

Ejemplo de etiqueta:

```
<input type="submit" value="TEXTO_VISIBLE"/>
```

Los ejemplos mostrados anteriormente, solo son una base para que tome idea del tipo de etiquetas HTML a manejar, la estilización y algunos otros parámetros dinámicos dependen de cómo desee presentar cada componente en el formulario.

Lenguaje de Reportería

Los usuarios que se hayan registrado en el sistema y que se encuentren logueados en la aplicación cliente, tendrán la posibilidad de ejecutar consultas hacia los datos registrados y recopilados por cada uno de los formularios a través de una estructura similar a las peticiones mostradas a lo largo de la descripción del proyecto. Sin embargo, deberá manejarse un lenguaje implícito llamado SQForm, que especificará la instrucción de consulta y acceso a datos que se debe manejar.

El formato de comandos de consulta es el siguiente.

```
SELECT TO FORM  ->  ID_FORMULARIO / NOMBRE
        [NOMBRE_CAMPO_1, NOMBRE_ CAMPO_2, NOMBRE_CAMPO_N, . . .]
WHERE [ ID / NOMBRE_CAMPO
         OPERADOR_RELACIONAL / LOGICO  VALOR_DE_COMPARACION . . .]
```

Donde:

- **SELECT TO FORM -> : Cláusula de proyección de información**
- **ID_FORMULARIO / NOMBRE : ID o NOMBRE del formulario a proyectar**
- **[. . . .] : Campos a proyectar**
- **WHERE [. . . .] : Cláusula de selección o restricción de información**
- **OPERADOR_RELACIONAL / LOGICO : Relacionales{<, >, <>, = ,>=, <= }, Lógicos{AND, OR, NOT}**
- **NOMBRE_CAMPO_1 : hace referencia al nombre de un campo en específico**
- **VALOR_DE_COMPARACION : hace referencia a un valor de comparación. Los valores numéricos deben ir sin comillas o apóstrofes, por ejemplo: 15, 10.5, etc. Las cadenas deben ir entre apóstrofes, por ejemplo: 'ejemplo_1', 'ejemplo_2'.**

Ejemplos de consultas:

La siguiente instrucción muestra todos los registros asociados al formulario $form_encuesta_1 :

```
SELECT TO FORM  ->  $form_encuesta_1  [ ]
```

La siguiente instrucción muestra los registros según los campos Cliente y Edad los cuales se encuentran asociados al formulario $form_encuesta_1 y cuyo campo Edad es mayor o igual a 18 y Cliente "Julio" :

```
SELECT TO FORM  ->  $form_encuesta_1
        [Cliente, Edad]
WHERE [ Edad
            >=  18
            AND Cliente = "Julio Paz" ]
```

Solicitud para reportes

Para enviar una consulta usando SQForm se debe utilizar el mismo formato de estructuración de solicitudes Indigo.

Nombre de la solicitud: **CONSULTAR_DATOS**
Parámetros:

- **CONSULTA-N*:** Consulta N en lenguaje SQForm.

Ejemplo:

```
<!ini_solicitud:"CONSULTAR_DATOS">
      { "CONSULTAS":[{
            "CONSULTA-1": " SELECT TO FORM -> $form_encuesta_1 [ ] ",
            "CONSULTA-2": " SELECT TO FORM -> $form_encuesta_1
                              [Cliente, Edad]
                            WHERE [ Edad
                                  >= 18
                                  AND Cliente = 'Julio Paz' ] ",
          }
        ]
      }
<fin_solicitud!>
```

Una vez enviada la solicitud al servidor el cliente debe esperar la respuesta ya sea con los datos obtenidos o con una descripción detallada de algún fallo que pudo ocurrir en la petición.

La respuesta del servidor debe ser basado también en el lenguaje Indigo, y debe contener los datos necesarios para armar uno o varios reportes según las consultas enviadas y el objetivo es que puedan desplegarse y visualizarse de forma amigable en el apartado que se mencionó al inicio en la aplicación cliente.

Estructura de almacenamiento de información

La estructura de almacenamiento de información, debe seguir la siguiente forma y debe contener mínimo la información siguiente de cada formulario para que se pueda volver a cargar una vez reiniciada la aplicación. Si se almacena en un solo archivo o en varios ya queda a criterio propio, además de ello se permite hacer pequeñas variaciones en cuanto a la manera de almacenar la estructura y datos de cada formulario, es decir si almacena aparte la estructura de los registros o cambia el orden de almacenamiento ya depende de su análisis y como se le haga más fácil, más **no se permite salirse**

completamente del estándar de estructuramiento que en este caso es JSON, ni tampoco es permitido reutilizar literalmente la misma estructura Indigo para almacenar datos.

Ejemplo de estructura de almacenamiento para formularios:

```
db.formularios(
  {
    "ID_FORMULARIO": "$id_form1",
    "TITULO": "Formulario para encuesta 1",
    "NOMBRE": "formulario_encuesta_1",
    "TEMA": "Dark",
    "USUARIO_CREACION": "User_Juanito"
     "ESTRUCTURA":(
            {
               "ID_COMPONENTE_1" : "$id_comp1",
               "PARAMETRO_1" :  "VALOR_PARAMETRO_1",
               …,
               "PARAMETRO_N" :  "VALOR_PARAMETRO_N",
            },
               …,
            {
               "ID_COMPONENTE_N" : "$id_comp_n",
               "PARAMETRO_1" :  "VALOR_PARAMETRO_1",
               …,
               "PARAMETRO_N" :  "VALOR_PARAMETRO_N",
            }
    ),
    DATOS_RECOPILADOS: (
            {
               "NOMBRE_CAMPO_1" : "Campo_1",
               "REGISTRO_1" :  "VALOR_REGISTRO_1",
               …,
               "REGISTRO_N" :  "VALOR_REGISTRO_N",
            },
               …,
            {
               "NOMBRE_CAMPO_N" : "Campo_N",
               "REGISTRO_1" :  "VALOR_REGISTRO_1",
               …,
               "REGISTRO_N" :  "VALOR_REGISTRO_N",
            }
    )
```

```
  },
  {
  ... otros formularios ...
  },
  ...
  ...
)
```

Importación y Exportación de Formularios

La importación de formularios sigue una estructura muy similar a la de almacenamiento, ya que prácticamente la idea es cargar la estructura de un formulario y ponerlo a disposición del usuario. Los archivos de entrada a importar deben ser con **extensión .form**, una vez cargado un archivo se debe tener un mecanismo como un botón que permita enviar la estructura como mensaje al servidor y que este se encargue de renderizar el formulario y almacenarlo.

Ejemplo de archivo:

```
new.formulario(
  {
    "ID_FORMULARIO": "$id_form1",
    "TITULO": "Formulario para encuesta 1",
    "NOMBRE": "formulario_encuesta_1",
    "TEMA": "Dark",
     "ESTRUCTURA":(
            {
               "ID_COMPONENTE_1" : "$id_comp1",
               "PARAMETRO_1" :  "VALOR_PARAMETRO_1",
               ...,
               "PARAMETRO_N" :  "VALOR_PARAMETRO_N",
            },
               ...,
            {
               "ID_COMPONENTE_N" : "$id_comp_n",
               "PARAMETRO_1" :  "VALOR_PARAMETRO_1",
               ...,
               "PARAMETRO_N" :  "VALOR_PARAMETRO_N",
            }
    )
  }
)
```

Para la exportación de formularios, se debe generar un archivo .form con la estructura antes descrita, y la idea es poder importarlos y cargarlos nuevamente para que estén a la disposición del usuario. Nota: únicamente se debe poder exportar la estructura del formulario.

Tomar en cuenta

- Todos los ID deben ser un conjunto de caracteres alfanuméricos incluyendo e iniciando con **_** , **-** o **$.**
- No se pueden repetir los ID de componentes asociados a un formulario ni tampoco ID de formularios.
- Los nombres de las solicitudes, y bloques de parámetros son case sensitive, en cambio los nombres de las etiquetas similares a XML no lo son.
- Todos los parámetros marcados con * son obligatorios.
- Se tomará en cuenta la estilización que le dé tanto a su sitio como a cada formulario, y sobre todo la usabilidad del sistema en conjunto.

Importante

- Lenguajes de programación válidos: JAVA SE, JSP, HTML, CSS, JavaScript.
- Usar herramientas Jflex y Cup para cualquier tipo de análisis lexico y sintactico.
- Proyecto obligatorio para tener derecho a la siguiente práctica.
- Las copias obtendrán nota de cero y se notificará a coordinación.
- Si se va a utilizar código de internet, entender la funcionalidad para que se tome como válido.

Entrega

La fecha de entrega es el día miércoles 31 de marzo a las 14:00 hrs. Los componentes a entregar son:

- Código fuente (enlace del repo de github)
- Ejecutables (jar, war, etc)
- Manual técnico incluyendo una sección con las expresiones regulares y las gramáticas usadas.
- Manual de usuario.

Calificación

Se calificará en la computadora de cada alumno por lo que es necesario que tengan instalado y configurado todo lo necesario.