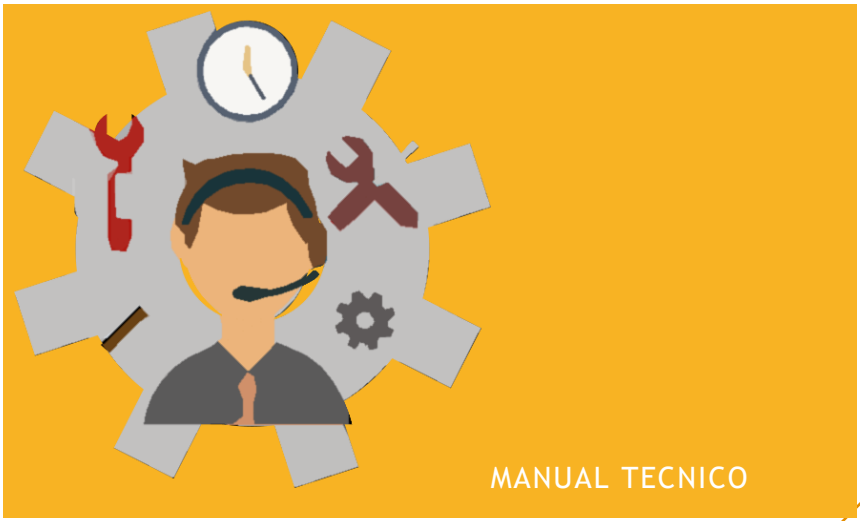
PROYECTO 2

ANTENNAE JAVA



Uzzi Libni Aaron Pineda Solorzano
201403541

Antes de modificar el software lea cuidadosamente este instructivo.

FUNCIONAMIENTO DE LA APLICACIÓN

La solución desarrollada cuenta con un tipo cliente-servidor de manera que esta se divide en las siguientes dos partes:

- **FRONTED:** Esta parte se encuentra desarrollada mediante el uso de javascript, css y html para la construcción de una pagina amigable con el cliente, esta pagina esta lanzada en un servidor htpp utilizando el lenguaje de Go.

- **BACKEND:** Esta parte esta desarrollada utilizando nodejs con javascript, para levantar un servidor que reciba peticiones REST, en esta parte se realiza el análisis léxico, sintactico y la detección de copias entre 2 proyectos desarrollados en java y de esta manera se obtienen reportes sobre el grado de similitud de los mismo. La herramienta a utilizar para el análisis léxico y sintáctico es jison.

REQUISITOS DEL SISTEMA

Esta aplicación puede utilizarse con los siguientes requerimientos:

1. Sistema operativo: Windows 10
2. Espacio En Disco Disponible: 1Gb
3. Memoria Ram: 8Gb.
4. Navegador Chrome, Microsoft Edge, Firefox

ANÁLISIS LÉXICO

Expresiones regulares utilizadas:

Er1 -> L(L|D|_)*
Er2 -> ;
Er3-> =
Er4-> ,
Er5-> +
Er6-> -
Er7-> *
Er8-> /
Er9-> && (And)
Er10-> || (or)

Er11-> ! (not)
Er12-> >(mayor)
Er13-> < (menor)
Er14-> >= (mayor o igual)
Er15-> <= (menor o igual)
Er16-> == (igual)
Er17-> != (distinto)
Er18-> (
Er19->)
Er20-> {
Er21-> }
Er22-> .
Er23-> "
Er24-> :
Er25-> D+(.D+)?

PALABRAS RESERVADAS UTILIZADAS

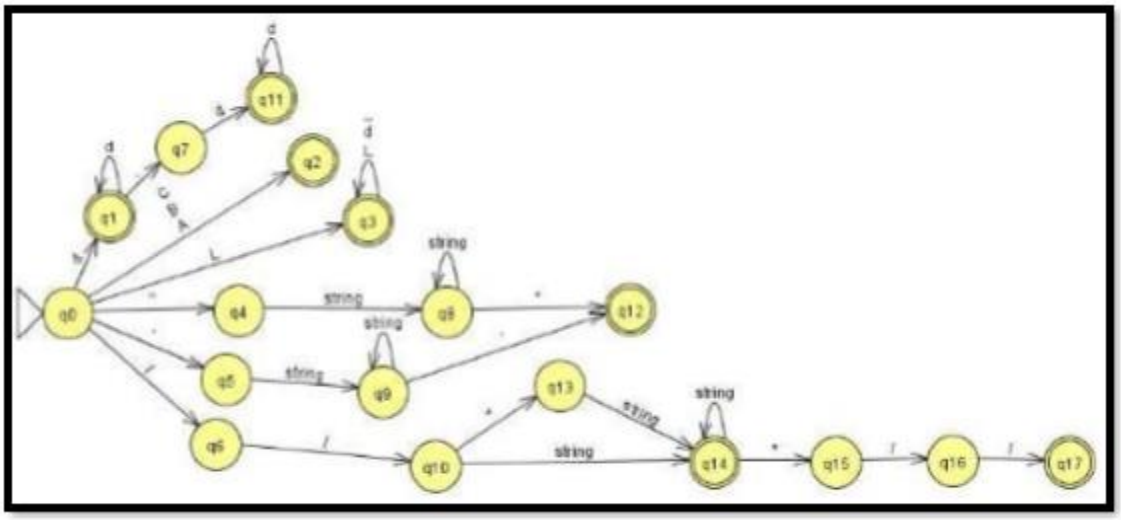
```
/*TIPO DE DATOS*/
"int"                   {return 'T_Int';}
"double"                {return 'T_Double';}
"boolean"               {return 'T_Boolean';}
"char"                  {return 'T_Char';}
"String"                {return 'T_String';}

/*CICLOS*/
"if"                    {return 'C_If';}
"else"                  {return 'C_Else';}
"for"                   {return 'C_For';}
"switch"                {return 'C_Switch';}
"case"                  {return 'C_Case';}
"default"               {return 'C_Default';}
"break"                 {return 'C_Break';}
"while"                 {return 'C_While';}
"do"                    {return 'C_DO';}
"return"                {return 'C_Return';}
```

```
/*PALABRAS RESERVADAS*/
"class"                 {return 'P_Class';}
"static"                {return 'P_Static';}
"void"                  {return 'P_Void';}
"main"                   {return 'P_Main';}
"args"                  {return 'P_Args';}
"public"                 {return 'P_Public';}
"private"                 {return 'P_Private';}
"protected"                 {return 'P_Protected';}
"this"                  {return 'P_This';}
"System"                   {return 'P_System';}
"out"                   {return 'P_Out';}
"println"                {return 'P_Println';}
"print"                 {return 'P_Print';}
"import"                  {return 'P_Import';}
"false"                 {return "P_False";}
"true"                  {return "P_True";}
"null"                  {return "P_Null";}
"continue"                 {return "P_Continue";}


/*SIMBOLOS*/
":"             {return 'S_Dospts';}
";"             {return 'S_Ptcoma';}
"{"             {return 'S_Llaveizq';}
"}"             {return 'S_Llaveder';}
"("             {return 'S_Parizq';}
")"             {return 'S_Parder';}
"["             {return 'S_Corizq';}
"]"             {return 'S_Corder';}
"."             {return 'S_Punto';}
","             {return 'S_Coma';}
```

```
/*OPERADORES LOGICOS*/
"&&"                                    {return 'OL_And'}
"&"                                        {return 'OL_Concat'}
"||"                                    {return 'OL_Or';}
"|"                                        {return 'OL_OR_OR';}
"!="                                        {return 'OL_Dif';}
"!"                                            {return 'OL_Not';}
"<="                        {return 'OL_Men_Ig'}
">="                        {return 'OL_May_Ig'}
"=="                        {return 'OL_Igual'}


/* OPERADORES RELACIONALES */
"<"                                            {return 'O_Men_que';}
">"                                            {return 'O_May_que';}
"="                                            {return 'O_Igual';}
"++"                        {return 'O_Inc'; }
"--"                       {return 'O_Decre'; }


/* OPERADORES ARITMETICOS */
"+"                                            {return 'A_Mas';}
"-"                                            {return 'A_Menos';}
"*"                                            {return 'A_Por';}
"/"                                            {return 'A_Dividido';}
"%"                        {return 'A_Modulo';}
"^"                        {return 'A_Potencia';}

/*  NUMEROS    */
[0-9]+("."[0-9]+)?\b                        {return 'Decimal';}
[0-9]+\b                                            {return 'Entero';}
```

```
/* IDENTIFICADORES */
([a-zA-Z_])[a-zA-Z0-9_]*                    {return 'Identificador';}

\s+                                         // se ignoran espacios en blanco
```

Este es el autómata generado para poder realizar el análisis léxico de el archivo de entrada.



ANÁLISIS SINTÁCTICO

Se utilizo la gramática siguiente:

```
//                                  GRAMATICA


%start INICIO


%%

// ---------------------------------------------- PRODUCCION INICIO ----------------------------
-----------------

INICIO

        : ESTRUCTURA EOF { console.log("Analisis Sintactico Concluido..."); return $1;}
```

```
;

//console.log(JSON.stringify($1, null, 2));


// ---------------------------------------------- PRODUCCION ESTRUCTURA ----------------------------------------

ESTRUCTURA

   : ESTRUCTURAP

   |

;


// ---------------------------------------------- PRODUCCION ESTRUCTURAP ----------------------------------------

ESTRUCTURAP

   : SINTAXISESTRUCTURA                          { $$ = [$1]; }

   | ESTRUCTURAP SINTAXISESTRUCTURA                  { $1.push($2); $$ = $1; }

;


// ---------------------------------------------- PRODUCCION SINTAXISESTRUCTURA ----------------------------------

SINTAXISESTRUCTURA

   : P_Import LISTAIMPORTS S_Ptcoma                                    { $$ = instruccionAPI.obtenerImport($2); }

   | P_Class Identificador S_Llaveizq INSTRUCCIONES_CLASE S_Llaveder          { $$ = instruccionAPI.obtenerClase($2,$4); }

   | error      { ESin.push({Tipo: 'SINTACTICO', Info: yytext, Linea: this._$.first_line, Columna: this._$.first_column}); }

;
```

```
//console.error('Error Sintáctico: ' + yytext + ', en la linea: ' + this._$.first_line + ', en la columna: ' + this._$.first_column);


// ---------------------------------------------- PRODUCCION LISTAIMPORTS ----------------------------

LISTAIMPORTS

    : Identificador                                 { $$ = [instruccionAPI.valor($1)]; }

    | LISTAIMPORTS S_Punto Identificador                 { $1.push(instruccionAPI.valor($3)); $$ = $1; }

;


// ---------------------------------------------- PRODUCCION INSTRUCCIONES_CLASE ----------------------------------

INSTRUCCIONES_CLASE

    : INSTRUCCIONES_CLASEP

    |

;


// ---------------------------------------------- PRODUCCION INSTRUCCIONES_CLASEP ----------------------------------

INSTRUCCIONES_CLASEP

    : INSTRUCCIONES_CLASEP LISTADO_INSTRUCCIONES_CLASE              { $1.push($2); $$ = $1; }   // $1 repite/vector $2 instruccion/nueva

    | LISTADO_INSTRUCCIONES_CLASE                               { $$ = [$1]; }  // $1 instruccion --> la vuelvo un vector

;
```

```
// ---------------------------------------------- PRODUCCION LISTADO_INSTRUCCIONES_CLASE ----------------------------------------------

LISTADO_INSTRUCCIONES_CLASE

    : P_Void Identificador S_Parizq PARAMETRO S_Parder S_Llaveizq INSTRUCCIONES S_Llaveder       { $$ = instruccionAPI.obtenerMetodo($2,$4,$7); }

    | TIPO_VAR Identificador S_Parizq PARAMETRO S_Parder S_Llaveizq INSTRUCCIONES S_Llaveder     { $$ = instruccionAPI.obtenerFuncion($1,$2,$4,$7); }

    | P_Void P_Main S_Parizq S_Parder S_Llaveizq INSTRUCCIONES S_Llaveder { $$ = instruccionAPI.obtenerMain($2,$6); }

    | TIPO_VAR DECLARACION S_Ptcoma                                                { $$ = instruccionAPI.obtenerDeclaracion($1, $2);  }

    | error { ESin.push({Tipo: 'SINTACTICO', Info: yytext, Linea: this._$.first_line, Columna: this._$.first_column}); }

;

// console.error('Error Sintáctico: ' + yytext + ', en la linea: ' + this._$.first_line + ', en la columna: ' + this._$.first_column);



// ---------------------------------------------- PRODUCCION PARAMETRO ----------------------------------------------

PARAMETRO

   : PARAMETROP

   |                                    { $$ = []; }

;


// ---------------------------------------------- PRODUCCION PARAMETROP ----------------------------------------------

PARAMETROP
```

```
   : TIPO_VAR Identificador                          { $$ =
[instruccionAPI.obtenerParametro($1,$2)]; }

   | PARAMETROP S_Coma TIPO_VAR Identificador              {
$1.push(instruccionAPI.obtenerParametro($3,$4)); $$ = $1; }

;


// ------------------------------------------- PRODUCCION TIPO_VAR ------------------------
---------------------

TIPO_VAR

   : T_Int

   | T_Double

   | T_Boolean

   | T_Char

   | T_String

;


// ------------------------------------------- PRODUCCION INSTRUCCIONES -------------------
----------------------------

INSTRUCCIONES

  : INSTRUCCIONESP

  |                                        { $$ = []; }

;


// ------------------------------------------- PRODUCCION INSTRUCCIONESP ----------------
-----------------------------

INSTRUCCIONESP

   : INFORMACION                                { $$ = [$1]; }
```

```
   | INSTRUCCIONESP INFORMACION                    { $1.push($2); $$ = $1; }

;


// ---------------------------------------------- PRODUCCION INFORMACION ----------------------------------------------

INFORMACION

// ---------------------------------------------- instruccion para declaracion de variables ----------------------------------------------

   : TIPO_VAR DECLARACION S_Ptcoma
{ $$ = instruccionAPI.obtenerDeclaracion($1, $2);  }

// ---------------------------------------------- instruccion para asignacion de variables ----------------------------------------------

   | Identificador O_Igual EXPRESION S_Ptcoma
{ $$ = instruccionAPI.obtenerAsignacion($1,[$3]); }




// ---------------------------------------------- instruccion condicion if ----------------------------------------------

   | C_If S_Parizq EXPRESION S_Parder S_Llaveizq INSTRUCCIONES S_Llaveder IFP
{ $$ = instruccionAPI.obtenerIF([$3],[$6],[$8]); }

// ----------------------------------------------instruccion condicion switch ----------------------------------------------

   | C_Switch S_Parizq EXPRESION S_Parder S_Llaveizq ESTRUCTURA_SWITCH
S_Llaveder                                    { $$ = instruccionAPI.obtenerSWITCH($3,$6);
}




// ---------------------------------------------- inicio de ciclos  for ----------------------------------------------
```

```
   | C_For S_Parizq SELECCION S_Ptcoma EXPRESION S_Ptcoma LISTAINCRE S_Parder
S_Llaveizq INSTRUCCIONES S_Llaveder              { $$ =
instruccionAPI.obtenerFOR($3,$5,$7,$10); }

// ------------------------------------------- ciclo while ------------------------------------
-------

   | C_While S_Parizq EXPRESION S_Parder S_Llaveizq INSTRUCCIONES S_Llaveder
{ $$ = instruccionAPI.obtenerWHILE($3,$6); }

// ------------------------------------------- ciclo do while ---------------------------------
-----------

   | C_DO S_Llaveizq INSTRUCCIONES S_Llaveder C_While S_Parizq EXPRESION
S_Parder S_Ptcoma                                { $$ =
instruccionAPI.obtenerDOWHILE($3,$7); }




// ------------------------------------------- instruccion para imprimir ----------------------
------------------------

   | P_System S_Punto P_Out S_Punto LISTAB S_Parizq EXPRESION S_Parder S_Ptcoma
{ $$ = instruccionAPI.obtenerImprimir([$7]); }

   | P_System S_Punto P_Out S_Punto LISTAB S_Parizq S_Parder S_Ptcoma
{ $$ = instruccionAPI.obtenerImprimir([]); }




// ------------------------------------------- palabras reservadas ---------------------------
-----------------

   | C_Break S_Ptcoma
{ $$ = [$1]; }

   | P_Continue S_Ptcoma
{ $$ = [$1]; }

   | C_Return RETURNP S_Ptcoma
{ $$ = [$2]; }
```

```
    | Identificador OPERADORES EXPRESION

    | error { ESin.push({Tipo: 'SINTACTICO', Info: yytext, Linea: this._$.first_line, Columna: this._$.first_column}); }

//   console.error('Error Sintáctico: ' + yytext + ', en la linea: ' + this._$.first_line + ', en la columna: ' + this._$.first_column);

;


LISTAB

   : P_Println

   | P_Print

;


LISTAINCRE

    : Identificador O_Igual EXPRESION

    | EXPRESION

;


// ------------------------------------------- PRODUCCION RETURNP ----------------------------------------

RETURNP

   : P_Null

   | Identificador O_Igual EXPRESION

   | Identificador

;
```

```
// ------------------------------------------- PRODUCCION DECLARACION ----------------------------------------

DECLARACION
    : LISTADO_DECLARACION                           { $$ = [$1]; }
    | DECLARACION S_Coma LISTADO_DECLARACION        { $1.push($3); $$ = $1; }
;


// ------------------------------------------- PRODUCCION LISTADO_DECLARACION ----------------------------------

LISTADO_DECLARACION
    : Identificador                                 { $$ = instruccionAPI.obtenerListaId($1, [{text: ''}]); }
    | Identificador O_Igual EXPRESION               { $$ = instruccionAPI.obtenerListaId($1, [$3]);}
;


// ------------------------------------------- PRODUCCION IFP ----------------------------------------

IFP
  : C_Else S_Llaveizq INSTRUCCIONES S_Llaveder                                  { $$ = $3; }
  | C_Else C_If S_Parizq EXPRESION S_Parder S_Llaveizq INSTRUCCIONES S_Llaveder IFP        { $$ = instruccionAPI.obtenerELSE_IF($4,$7); }
  |                                                                             { $$ = []; }
;


// ------------------------------------------- PRODUCCION ESTRUCTURA_SWITCH ----------------------------------
```

```
ESTRUCTURA_SWITCH

      : ESTRUCTURA_SWITCH LISTACASO        { $1.push($2); $$ = $1;}

      | LISTACASO                          { $$ = instruccionAPI.obtenerCASOS($1);}

;


// ---------------------------------------------- PRODUCCION LISTACASO -----------------------
----------------------

LISTACASO

   : C_Case EXPRESION S_Dospts INSTRUCCIONES                        { $$ =
instruccionAPI.obtenerNUEVOCASO($2,$4); }

   | C_Default S_Dospts INSTRUCCIONES                               { $$ =
instruccionAPI.obtenerDEFAULT($3); }

;


// ---------------------------------------------- PRODUCCION SELECCION -----------------------
-----------------------

SELECCION:

   | TIPO_VAR DECLARACION

   | Identificador O_Igual EXPRESION

;


// ---------------------------------------------- PRODUCCION EXPRESION -----------------------
-----------------------

EXPRESION

// ---------------------------------------------- Expresiones para cadenas -------------------
-------------------------

    : EXPRESION OL_Concat EXPRESION                    { $$ =
instruccionAPI.obtenerOpBinaria($1,$3,TIPOOPERACION.OPCONCAT); }
```

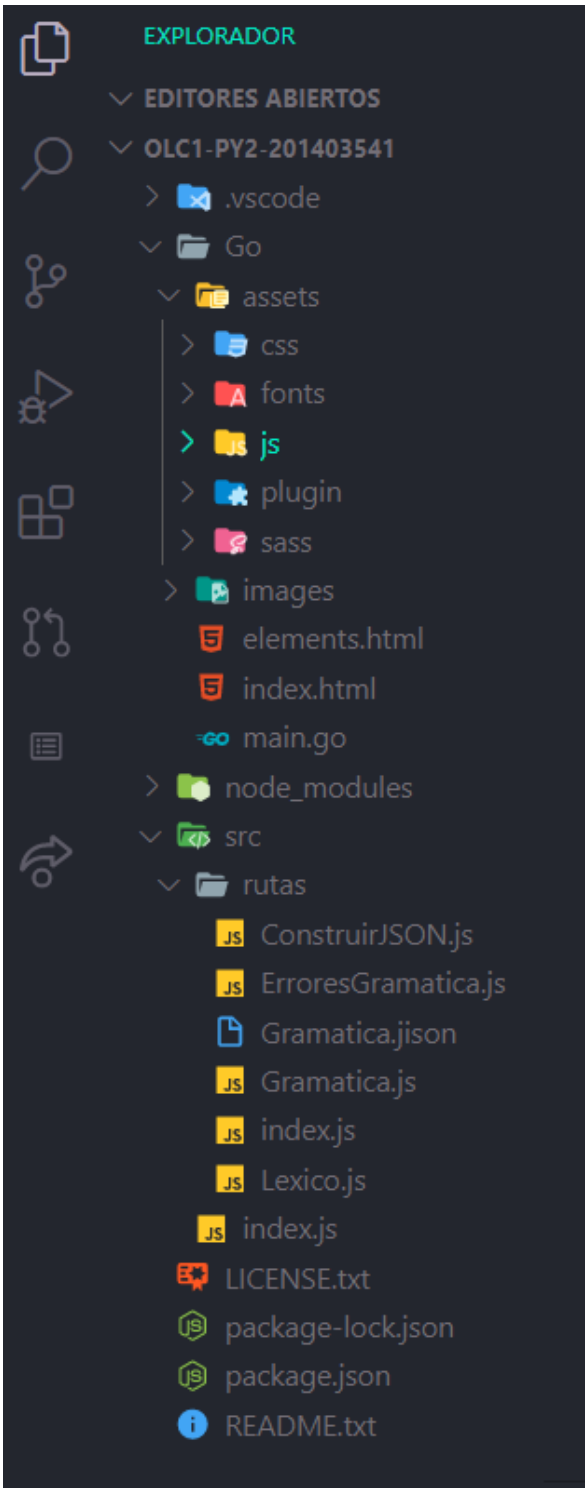
```
// ---------------------------------------------- Expresiones para numeros ----------------------
------------------------

    | A_Menos EXPRESION %prec starwars                  { $$ =
[instruccionAPI.obtenerOpUnaria($1,TIPOOPERACION.OPNEG)]; }

    | EXPRESION A_Mas EXPRESION                       { $$ =
[instruccionAPI.obtenerOpBinaria($1,$3,TIPOOPERACION.OPSUMA)]; }

    | EXPRESION A_Menos EXPRESION                     { $$ =
[instruccionAPI.obtenerOpBinaria($1,$3,TIPOOPERACION.OPRESTA)]; }

    | EXPRESION A_Dividido EXPRESION                  { $$ =
[instruccionAPI.obtenerOpBinaria($1,$3,TIPOOPERACION.OPDIV)]; }

    | EXPRESION A_Por EXPRESION                       { $$ =
[instruccionAPI.obtenerOpBinaria($1,$3,TIPOOPERACION.OPMULTI)]; }

    | EXPRESION A_Potencia EXPRESION                  { $$ =
[instruccionAPI.obtenerOpBinaria($1,$3,TIPOOPERACION.OPPOT)]; }

    | EXPRESION A_Modulo EXPRESION                    { $$ =
[instruccionAPI.obtenerOpBinaria($1,$3,TIPOOPERACION.OPMOD)]; }

    | S_Parizq EXPRESION S_Parder                     { $$ = [$2]; }


// ---------------------------------------------- Expresiones relacionales ----------------------
----------------------

    | EXPRESION O_Men_que EXPRESION                   { $$ =
instruccionAPI.obtenerOpBinaria($1,$3,TIPOOPERACION.OPMENORQUE); }

    | EXPRESION O_May_que EXPRESION                   { $$ =
instruccionAPI.obtenerOpBinaria($1,$3,TIPOOPERACION.OPMAYORQUE); }

    | EXPRESION OL_May_Ig EXPRESION                   { $$ =
instruccionAPI.obtenerOpBinaria($1,$3,TIPOOPERACION.OPMAYORIGUAL); }

    | EXPRESION OL_Men_Ig EXPRESION                   { $$ =
instruccionAPI.obtenerOpBinaria($1,$3,TIPOOPERACION.OPMENORIGUAL); }
```

```
    | EXPRESION OL_Igual EXPRESION                    { $$ =
instruccionAPI.obtenerOpBinaria($1,$3,TIPOOPERACION.OPEQUALS); }

    | EXPRESION OL_Dif EXPRESION                     { $$ =
instruccionAPI.obtenerOpBinaria($1,$3,TIPOOPERACION.OPDIF); }




// ---------------------------------------------- Expresiones logicas ----------------------------
-----------------

    | EXPRESION OL_And EXPRESION                     { $$ =
instruccionAPI.obtenerOpBinaria($1,$3,TIPOOPERACION.OPAND); }

    | EXPRESION OL_Or EXPRESION                     { $$ =
instruccionAPI.obtenerOpBinaria($1,$3,TIPOOPERACION.OPOR); }

    | OL_Not EXPRESION %prec starwars               { $$
=instruccionAPI.obtenerOpUnaria($2,TIPOOPERACION.OPNOT); }


// ---------------------------------------------- Otras instrucciones relacionadas con
expresiones ----------------------------------------------

    | TIPO_DATO O_Decre %prec starwars               { $$ =
instruccionAPI.obtenerOpUnaria($2,TIPOOPERACION.OPDECRE); }

    | TIPO_DATO O_Inc %prec starwars                 { $$ =
instruccionAPI.obtenerOpUnaria($2,TIPOOPERACION.OPINCRE); }

    | TIPO_DATO

    | Identificador S_Parizq LISTAEXPRESION S_Parder

;


// ---------------------------------------------- PRODUCCION TIPO_DATO ----------------------
----------------------

TIPO_DATO
```

```
    : Entero                                    { $$ =
instruccionAPI.obtenerValor($1,TIPODATO.TNUMERO); }

    | Decimal                                    { $$ =
instruccionAPI.obtenerValor($1,TIPODATO.TDECIMAL); }

    | P_False                                   { $$ =
instruccionAPI.obtenerValor($1,TIPODATO.TFALSE); }

    | P_True                                    { $$ =
instruccionAPI.obtenerValor($1,TIPODATO.TTRUE); }

    | Cadena                                    { $$ =
instruccionAPI.obtenerValor($1,TIPODATO.TCADENA); }

    | Char                                     { $$ =
instruccionAPI.obtenerValor($1,TIPODATO.TCHAR); }

    | Identificador                             { $$ =
instruccionAPI.obtenerValor($1,TIPODATO.TIDENTIFICADOR); }

;


// ------------------------------------------- PRODUCCION OPERADORES ----------------------
-------------------------

OPERADORES

   : A_Mas                { $$ =
instruccionAPI.obtenerOperador(TIPOOPERACION.OPSUMA); }

   | A_Menos              { $$ =
instruccionAPI.obtenerOperador(TIPOOPERACION.OPRESTA); }

   | A_Por                { $$ =
instruccionAPI.obtenerOperador(TIPOOPERACION.OPMULTI); }

   | A_Dividido           { $$ =
instruccionAPI.obtenerOperador(TIPOOPERACION.OPDIV); }

   | A_Potencia           { $$ =
instruccionAPI.obtenerOperador(TIPOOPERACION.OPPOT); }
```

```
    | A_Modulo                { $$ =
instruccionAPI.obtenerOperador(TIPOOPERACION.OPMOD); }

;


// ---------------------------------------------- PRODUCCION LISTAEXPRESION -----------------
------------------------------

LISTAEXPRESION

    : LISTAEXPRESIONP

    |                                   { $$ = []; }

;


// ---------------------------------------------- PRODUCCION LISTAEXPRESIONP ---------------
------------------------------

LISTAEXPRESIONP

    : EXPRESION                             { $$ = [$1];}

    | LISTAEXPRESIONP S_Coma EXPRESION          { $1.push($3); $$ = $1;}

;
```

ESQUEMA



Visualización de las carpetas utilizadas en el proyecto. Carpeta utilizada para el FRONTEND Go, carpeta utilizada para el BACKEND src.