Compiler2Project

اعداد الطلاب :

1- محمد مصطفى لقيس
2 - محمد عامر العيساوي
3 - أحمد ديش
4- مهند البغدادي
5- بتول عيسى
6- ديانا عسكر

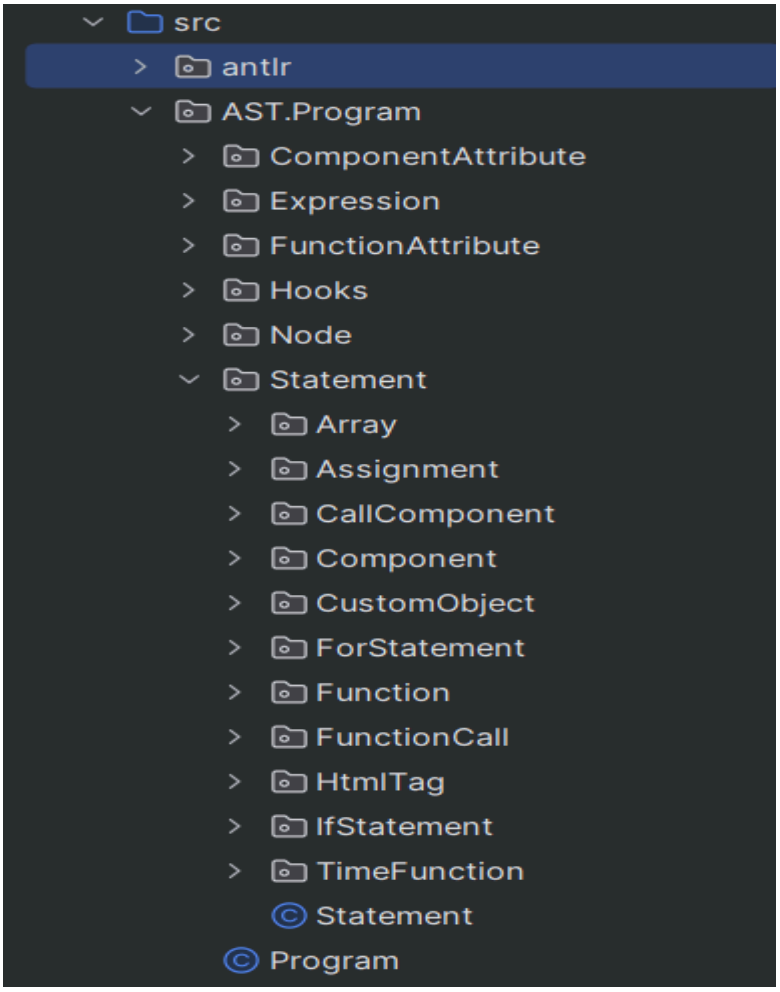
بنية المشروع:

المشروع موجود ضمن المجلد Group 51 \ Code \ src ويحتوي على :

1- Antlr
2- Ast
3- Final code
4- GenerateCode
5- SemanticCheck
6- SymbolTbale
7- Visitor

المجلد antlr : يحتوي عل ال lexer وهو ReactLexer.g4 و ال parser وهو ReactParser.g4.

المجلد AST : يحتوي على مجلد رئيسي program والذي يحوي الصفوف المستخدمة لبناء ال AST حيث كل صف موجود ضمن مجلد خاص مع الصفوف المستخدمة ضمنه بشكل شجري.



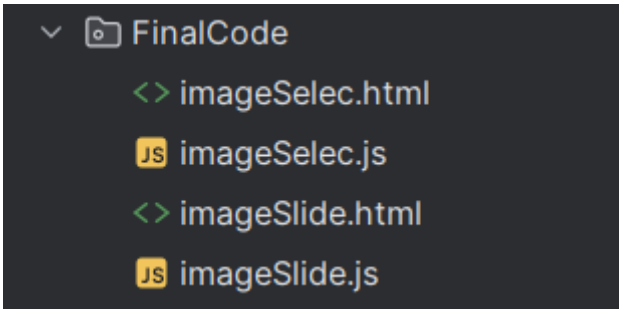
المجلد FinalCode : يحتوي على الكود الذي سوف يتولد في النهاية عند التنفيذ حيث سوف يتم توليد ملفين نصيين وذلك لكل برنامج اختبار والملفين هما :

1- اسم الملف.html

2-اسم الملف.js

حيث اسم الملف هو الاسم الذي اقوم بوضعه بنفسي تعبيرا عن البرنامج الذي أقوم ببنائه.

مثال :



المجلد GenerateCode: الصف المسؤول عن عملية ال codeGeneration حيث يحتوي على :

-**ProjectName:**

اسم الملف الذي سأقوم بوضعه ضمن المجلد FinalCode كما قمنا بالتوضيح سابقا.

-**FileWriterJs:**

القارئ الذي من خلاله يمكنني انشاء الملفات والكتابة ضمنها.

-**generate:**

هذا التابع يأخذ بارمترين هما :

symbolTable : جدول الرموز والذي من خلاله سوف أحصل على اسم الملف لأقوم بوضعه في مجلد FinalCode.

statements : البيانات التي سوف اقوم بكتابتها ضمن ملف ال java script.

ثم يقوم هذا التابع بالمرور على ال symbolTable للحصول على اسم الملف وتخزينه في projectName, ثم يقوم باستدعاء التوابع الثلاثة التالية.

-**generateHtmlFile:**

هذا التابع يقوم بتوليد ملف ال html ويقوم بالكتابة ضمنه.

-**generateJsFile:**

هذا التابع يقوم بانشاء ملف java script.

- **writeJsFile:**

هذا التابع يقوم بالمرور على ال statements ثم يقوم بكتابتها ضمن ال ملف java script.

-**formatJsFile:**

هذا التابع يقوم بحذف التكرارات من ملف ال java script.

المجلد SemanticCheck: يحتوي على الصف المسؤول عن عملية التحقق النحوي ويحتوي على اربع توابع للتحقق هذه التوابع تأخذ بارامترات وهي (جدول الرموز , رقم السطر , البيانات التي سأقوم بفحصها) وهي :

-**checkIfIdentifierNotExist:**

هذا التابع يقوم بالتحقق فيما إذا كان المتغير غير موجود ضمن الكود الخاص بي وبالتالي عند استدعاء المتغير او استخدامه سيقوم برد خطأ يوضح فيه أن المتغير غير معرف في الكود.

-**checkIfComponentNotExist:**

هذا التابع يقوم بالتحقق فيما إذا كان المكوّن غير موجود ضمن الكود الخاص بي وبالتالي عند استدعاء المكوّن او استخدامه سيقوم برد خطأ يوضح فيه أن المكوّن غير معرف في الكود.

-**checkIfIdentifierIsDefined:**

هذا التابع يقوم بالتحقق فيما إذا كان المتغير معرّف مسبقا ضمن الكود الخاص بي وبالتالي عند اعادة تعريفه سيقوم برد خطأ يوضح فيه أن التابع معرّف مسبقا.

-**checkIfComponentIsDefined:**

هذا التابع يقوم بالتحقق فيما إذا كان المكوّن معرّف مسبقا ضمن الكود الخاص بي وبالتالي عند اعادة تعريفه سيقوم برد خطأ يوضح فيه أن المكوّن معرّف مسبقا.

-**checkIfHooksIsOutside:**

هذا التابع يقوم بالتحقق فيما إذا كان ال [useState , useEffect , useRef] : hooks معرّفة خارج حدود المكوّن.

المجلد SymbolTable: يحتوي على صفين وهما :

-**Row:**

السطر الذي يتكون منه جدول الرموز ويحتوي هذا السطر على :

-**String Type :** اسم الرمز
-**String Value :** البيانات المخزنة
-**int Line :** رقم السطر الذي يحوي البيانات

-**SymbolTable:**

جدول الرموز ويتكون من List of Row.

مجلد Visitor: يحتوي على الصف Visitor وهو الصف الذي يحوي كل التوابع المسؤولة عن تعبئة شجرة ال AST بشكل مترابط ويحوي ايضا :

-**List CallComponentName:**

قائمة مساعدة تحتوي على المكونات التي تم استدعائها ضمن الكود الخاص بي حتى اتمكن من التحقق النحوي عن طريق التابع **checkIfComponentNotExist** .

-**List ComponentName:**

قائمة مساعدة تحتوي على تعريف المكونات حتى اتمكن من التحقق النحوي عن طريق التابع **checkIfComponentIsDefined**.

-**symbolTable :**

هذا الكائن لتعبئة جدول الرموز من خلال التوابع الموجودة ضمن الصف Visitor .

-**generateCode:**

هذا الكائن لتوليد اللغة المستهدفة.

-**statements:**

هذه القائمة تحتوي على البيانات النهائية للكود الخاص بي.

التوابع :

أهمهم تابع **visitProgram** وهو التابع الذي يبدأ في زيارة التوابع المتبقية بالتسلسل ليقوم بتعبئة شجرة ال AST , ايضا في هذا التابع اقوم بالتحقق النحوي ل (المكونات , المكونات المستدعاة)وذلك بعد الانتهاء من تعبئة الشجرة من خلال استدعاء التابعين :

checkIfComponentNotExist , **checkIfComponentIsDefined**

أيضا في هذا التابع اقوم باستدعاء التابع generate من خلال الكائن المعرف مسبقا generateCode واقوم بتمرير جدول الرموز والبيانات النهائية للكود الخاص بي حتى يقوم بتوليد اللغة الهدف.

ثم اقوم بطباعة جدول الرموز حتى يظهر لنا على الخرج وايضا اقوم بارجاع الكائن program حتى اقوم بطباعة شجرة ال AST وذلك في التابع الرئيسي main.

خوارزمية التحويل للغة الهدف :

في كل صف من صفوف ال AST اقوم بتعريف تابع generateJS وهذا التابع يقوم ببناء اللغة الهدف لهذا الصف

وعند استدعاء التابع generate في التابع program الموجود ضمن توابع ال Visitor فان هذا التابع يقوم بالمرور على كل بيانات الكود الخاص بي واستدعاء التابع generateJs الموجود ضمن كل واحدة من هذه البيانات.