**Back-End Requirements**

1. Parser
2. Library
3. Linker
4. Error Detection
5. Symbol Table

I. Parser

**A - Parser Objectives**

* The parser is to handle the reading of the input from the built in text editor when the assembler is invoked. After confirming correct input data, the parser is to look up the input in the symbol table and use the library functions to perform the action specified by the input if necessary. The parser will receive a line of input from the linker that is to be parsed and access to the symbol table. The parser will return any error codes that are triggered by the library and error codes that occur from incorrect input

**B - Parser Requirements**

* Check input for valid data and incorrect formatting
* Access the symbol table and update it accordingly
* Call the library to determine action to be used on the input data
* Return success or failure to the linker from the parser
* Return success or failure

II. Library

**A - Library Objectives**

* The library is to maintain a record of all recognized commands from ASSIST/I in accordance with the commands specified in the specification documentation. The library will take input from the Parser in the form of command tokens and argument tokens. The library will return a success code to the linker as defined by the error detection documentation.

**B - Library Requirements**

* Check input from the parser with the ASSIST/I definitions in memory
* Maintain a valid ASSIST/I definition bank to perform actions based on the command given

III. Linker

**A - Linker Objectives**

* The linker will act as a sort of back-end driver, providing the connections between the back-end components and connecting the information from the back-end to the front-end. The linker will be the portion of the back-end that calls the other parts of the back-end and enables them to work together. The linker will be responsible for the symbol table and all updating of it. The linker will need to use the information it receives from the back-end structures to determine the correct course of action. The linker will need to use the error codes received by the back-end structures and send them to another back-end structure designed to handle error codes.

**B – Linker Requirements**

* Provide a framework of function calls to back-end structures
* Maintain symbol table
* Initialize symbol table
* Have access to the GUI to make updates as needed
* Connect the back-end structures to the error handling structure

1. Error Detection

**A – Error Detection Objectives**

* The error detection will take as input a set of predetermined error codes that match with a certain type of error found in ASSIST/I. The error detection will need to print an error message onto the output screen that matches with the error that was thrown.

**B – Error Detection Requirements**

* Maintain a listing of errors associated with certain error codes
* Print a message detailing the type of error

V. Symbol Table

**A – Symbol Table Objectives**

* The symbol table is a source listing of all acceptable “tokens” that can be found in the language. The symbol table will also keep record of labels and hold a memory address where the label is referenced in the program. The symbol table will use a hash function to quickly find and add aditional entries into the table.

**B – Symbol Table Requirements**

* Maintaing a listing of all acceptable “tokens” of the ASSIST/I language used in ASSIST/UNA
* Hash table that stores records of valid identifiers
* Provide feedback upon finding/not finding a matching entry