PROJECT REPORT

LEX FILE:

I have declared different types of tokens for the corresponding HTML tags so that during lexical analysis it can identify the tokens and if some unrecognized character appears in the input file, lex file will not recognize that and it will throw an error. It can identify some of the greek characters and special characters as well. It will simply ignore the comment if it appears anywhere in the input file. For all the given correct input, tokens will be generated and out first stage of compilation is done. We have our desired tokens, now they would be used for further stage of compilation.

YACC FILE :

Yacc or Bison file is using tokens generated after the lexical analysis of the inputs. The grammar of our parser is defined in the bison file. We are generating the possible regualar expression with the help of the tokens. Even if the tokens are correct, there is no guarantee that the expression of the input is correct. It basically does the syntax analysis of our given input which were converted into the tokens. If it is syntactically correct, it will proceed further else it will throw an error because of being syntactically incorrect. After syntax analysis it goes into next step of parsing.

AST(Abstract syntax tree)

I am storing the following data in every nodes of the AST.

```
typedef struct node{
        node_type t;
        string value;
       int k;
        vector<node*> child;
}node;
```

There is node_type which conatains what kind of node it is like (n_headings, n_table) means node containing data of heading will be of type n_headings, node containig data of table is of type n_table. There are many more like these. It also contains a value associated with it, that we can use to store something in the node. There is one integer variable associated with a node that we will be using for the purpose of traversal, it could have been boolean or any other datatype. This is just for the sake of traversal. Then there is a vector of type node* which will store the children of the nodes that we can use to find the while information of the tree. It will help in traversal as well. Any node can have any number of children, it's not like the simple binary tree.

I have transalated the AST in a way like when the lexical analysis was being done, i have stored the corresponding latex replacement of the html tags and storing them in the nodes of the tree. So, when traversing we will get the desired latex output of the corresponding html file.

I have used the language C++ for making the lexical file and parser file.

Extra things i have done in the assignment:

1. I have handled the greek words which wasn't mentioned in the assignment.
2. I have handled many special characters as well.
3. I am storing the type of nodes in the node of abstraxct syntax tree so as for better understanding.