Week 2

Week2

Main Goal

- Find API methods related to ReactJs
- Design queries to collect the dataset
- Analyse the results and visualize them

API on Documentation

```
Please see React API on Web Page
```

Code based API methods

```
  <!-- ... other HTML ... -->

  <!-- Load React. -->
  <!-- Note: when deploying, replace "development.js" with "production.min.js". -->
  <script src="https://unpkg.com/react@16/umd/react.development.js" crossorigin></script>
  <script src="https://unpkg.com/react-dom@16/umd/react-dom.development.js" crossorigin>
</script>

  <!-- Load our React component. -->
  <script src="like_button.js"></script>

</body>
```

Result

```
Please see ReactAPIFunctions extracted from Source Code
```

Code:

```
import java.io.*;
import java.util.Scanner;
import java.util.TreeSet;

public class ReactAPI {
    public static TreeSet<String> functions = new TreeSet<>();
    public static boolean isWord(char c){
        if(c>='a' && c<='z' || c>='A' && c<='Z'){
            return true;
        }
        return false;
    }

    public static void handleFile(String filename) throws FileNotFoundException {
        File js1 = new File(filename);
        BufferedReader reader = null;
        reader = new BufferedReader(new FileReader(js1));
        Scanner in = new Scanner(reader);
        while (in.hasNextLine()){
            String str = in.nextLine();
//            System.out.println("Line:"+str);
            if(str.contains("function")){
                int posi = str.indexOf("function");
//                int last = str.lastIndexOf("function");
//                if(posi!=last)System.out.println("ERROR");//YES!!! It appears.
                Scanner sin = new Scanner(str);
                while (sin.hasNext()){
                    String token = sin.next();
                    if(token.equals("function")){
                        if(!sin.hasNext())break;
                        token=sin.next();
                        if(token.contains("(")){
                            String name = token.substring(0,token.indexOf('('));
                            if(name.length()!=0){
                                functions.add(name);
                                System.out.println("ADD:"+name);
                            }
                        }
                    }
                }
//                int nxtToken = posi+9;
//                String name = "";
//                int sl = str.length();
//
//                while (isWord(str.charAt(nxtToken))){
//                    name = name+str.charAt(nxtToken);
//                    nxtToken++;
//                    if(nxtToken>=sl){
//                        name="";
//                        break;
//                    }
//                }
//                if(!name.equals("") && str.charAt(nxtToken)=='('){
//                    functions.add(name);
//                    System.out.println("ADD:"+name);
//                }


            }
        }
```

```
    }

    public static void main(String args[]) throws IOException {
        functions.clear();
        try {
//            handleFile("test.txt");
            handleFile("react.development.js");
            handleFile("react-dom.development.js");
        } catch (FileNotFoundException e) {
            e.printStackTrace();
        }
        File output = new File("ReactAPIFunctions.txt");
        FileWriter fw = null;
        fw = new FileWriter(output);
        for(String str:functions){
            System.out.println(str);
            fw.write(str+"\r\n");
        }
        fw.flush();
        fw.close();

    }

}
```

Final API Methods in Research

Please see React API on Web Page – We currently focus on

Queries

NO.1

```
select * from Posts p
where
p.Tags like '%##key##%' and
p.Title like '%##key##%' and
p.Body like '%##key##%' and
p.Body like '%##Method##%'
```

NO.2

```
select count(*) from Posts p
where
p.Tags like '%##key##%' and
(p.Title like '%##key##%' or p.Title like '%##Method##%') and
p.Body like '%##key##%' and p.Body like '%##Method##%' and
( p.Body like '%js%'or p.Body like '%JS%')
```

NO.3

```
select count(*) from Posts p
where
p.Tags like '%##key##%' and
(p.Title like '%##key##%' or p.Title like '%##Method##%') and
p.Body like '%##key##%' and p.Body like '%##Method##%'
```

NO.4

```
select count(*) from Posts p
where
(p.Tags like '%##key##%' or p.Tags like '%##Method##%') and
(p.Title like '%##key##%' or p.Title like '%##Method##%') and
p.Body like '%##key##%' and p.Body like '%##Method##%'
```

NO.5 Might Useful

```
select count(*) from Posts p
where
(p.Tags like '%##key##%' or p.Tags like '%##Method##%') and
(p.Title like '%##key##%' or p.Title like '%##Method##%') and
p.Body like '%##key##%' and p.Body like '%##Method##%' and
( p.Body like '%js%'or p.Body like '%JS%')
```

NO.6 Data Collected – With Tag and 'JS' Constrain

```
select * from Posts p
where
p.Tags like '%##key##%' and
(p.Title like '%##key##%' or p.Title like '%##Method##%') and
p.Body like '%##key##%' and p.Body like '%##Method##%' and
( p.Body like '%js%'or p.Body like '%JS%')
```

NO.7 Data Collected – No Tag Constrain Data rows

```
select * from Posts p
where
(p.Title like '%##key##%' or p.Title like '%##Method##%') and
p.Body like '%##key##%' and p.Body like '%##Method##%' and
( p.Body like '%js%'or p.Body like '%JS%')
```

NO.8 Data Collected – No JS and Tag Constrain

```
select * from Posts p
where
(p.Title like '%##key##%' or p.Title like '%##Method##%') and
p.Body like '%##key##%' and p.Body like '%##Method##%'
```