PROBLEM DEFINITION:

- Given a value in a JSON need to find out all the parents and the children
- Given a value in JSON need to find out all paths in a undirected graph.
- Preferred language – JavaScript.

SOLUTION:

ALGORITHM:

INPUT:

(1) **filter-json.json –** File containing menu in JSON format
(2) **Name of node –** Entered in text box for output

OUTPUT:

(1) **List of Ancestors** – List of nodes from parent to entered input node
(2) **List of Descendants** – List of all nodes from current to leaf nodes

STEPS:

1) Input the JSON file into the page using the controls provided. {1}{2}
2) Receive the file and create a JavaScript object file "obj" to manipulate the imported file. {1}
3) Receive the text box input as name of node whose ancestors and descendants are displayed.
4) Traverse obj recursively to find out which branch from the root the given input node belongs to. {3}
5) When the input node is found, throw the current version of obj using a user defined exception to handle the display of all descendants separately. {2}
6) Using the object thrown, all its descendants are recursively added to the path string. {3}
7) Provide proper indentation for ancestor and descendant display.
8) The path string is used to fill the ancestor and descendant lists separately.

PSEUDO CODE:

```
START
CALL handleFileSelect and hence receivedText to get JSON file input {1}{2}
Parse the JSON file to a JavaScript object "obj" {1}
Receive name of node input and proceed if it is a genuine node name
CALL find with obj and the name of node
IF input supplied is present in the path string
        Display ancestors and descendants from the path string
ELSE
        Display error message about bad input
ENDIF
END
```

```
FUNCTION find
        PASS IN: JavaScript Object for JSON file and the Input name of node
        BEGIN
                CALL recursive with obj and input node
        EXCEPTION                                                          {2}
                WHEN obj type is thrown
                        CALL desc with obj caught by exception
        END
        PASS OUT: The path string
ENDFUNCTION

FUNCTION recursive
        PASS IN: The JavaScript object “obj” and input name of node        {3}
        FOR all keys present in obj
                IF type of obj[key] is object
                        Add key to path string
                        CALL recursive with obj[key] and input name of node
                        Remove the key attached to the path string
                        IF key is the input name of node
                                THROW obj
                        ENDIF
                ELSE
                        Add new leaf node by removing already existing leaf node if any
                ENDIF
        ENDFOR
        PASS OUT: The path string
ENDFUNCTION

FUNCTION desc
        PASS IN: The JavaScript object for JSON file at input node depth   {3}
        FOR all keys present in obj
                IF obj[key] is a leaf node
                        Add all possible keys of obj to path
                ELSE
                        Add the immediate descendant to the path string
        ENDFOR
        PASS OUT: The path string
ENDFUNCTION
```

DESCRIPTION:

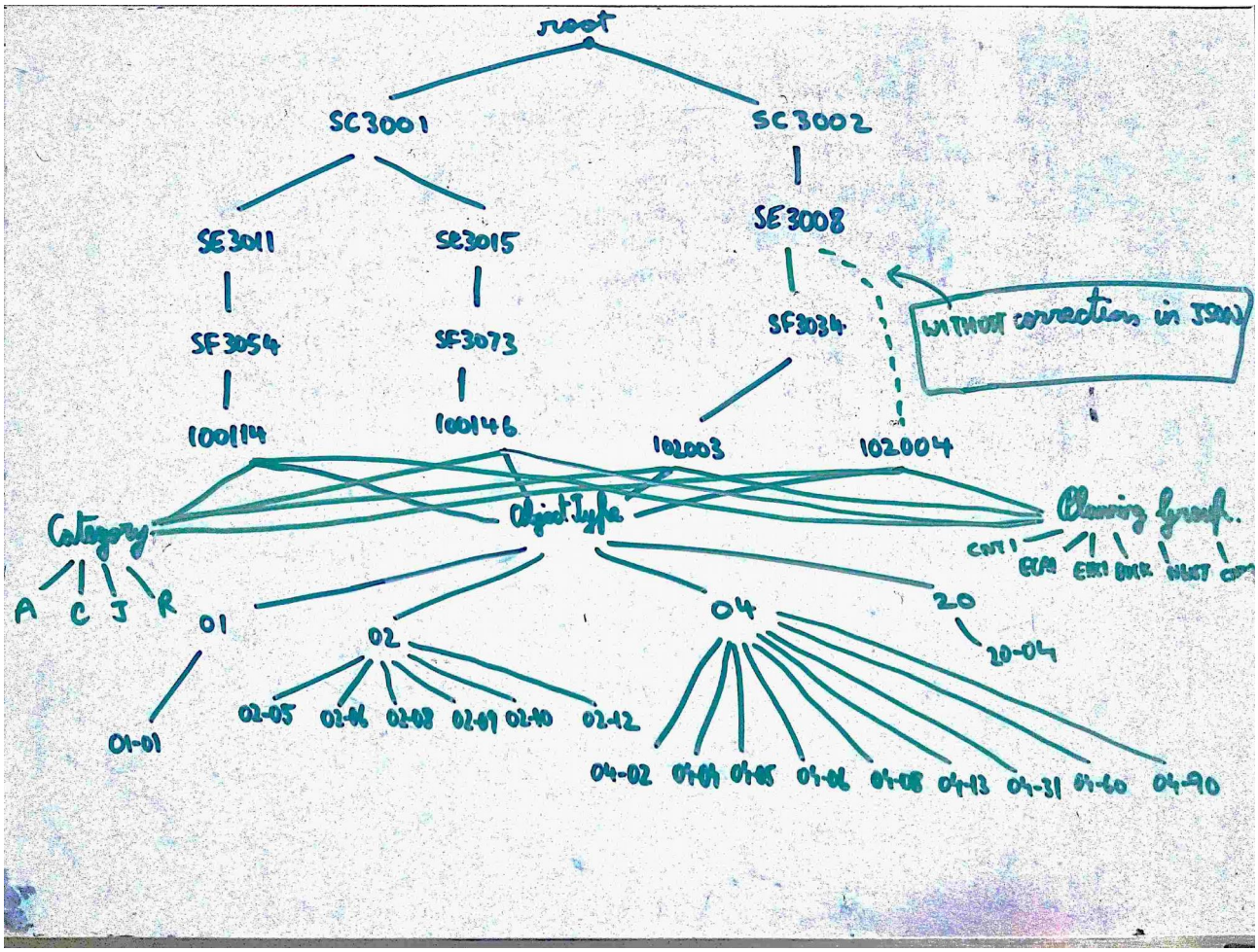
1. The JSON file is accepted as input and parsed into a JavaScript object.
2. The object is then recursively traversed by a recursive DEPTH FIRST SEARCH strategy, while storing the path traversed into a string, until the input node is found.
3. Once input node is found, an exception is thrown using the object at its current depth.
4. That object is then used for a similar DFS traversal to add all child nodes to the same path string.
5. The path string is then properly indented according to the use case specified.

IMPLEMENTATION TECHNOLOGY:

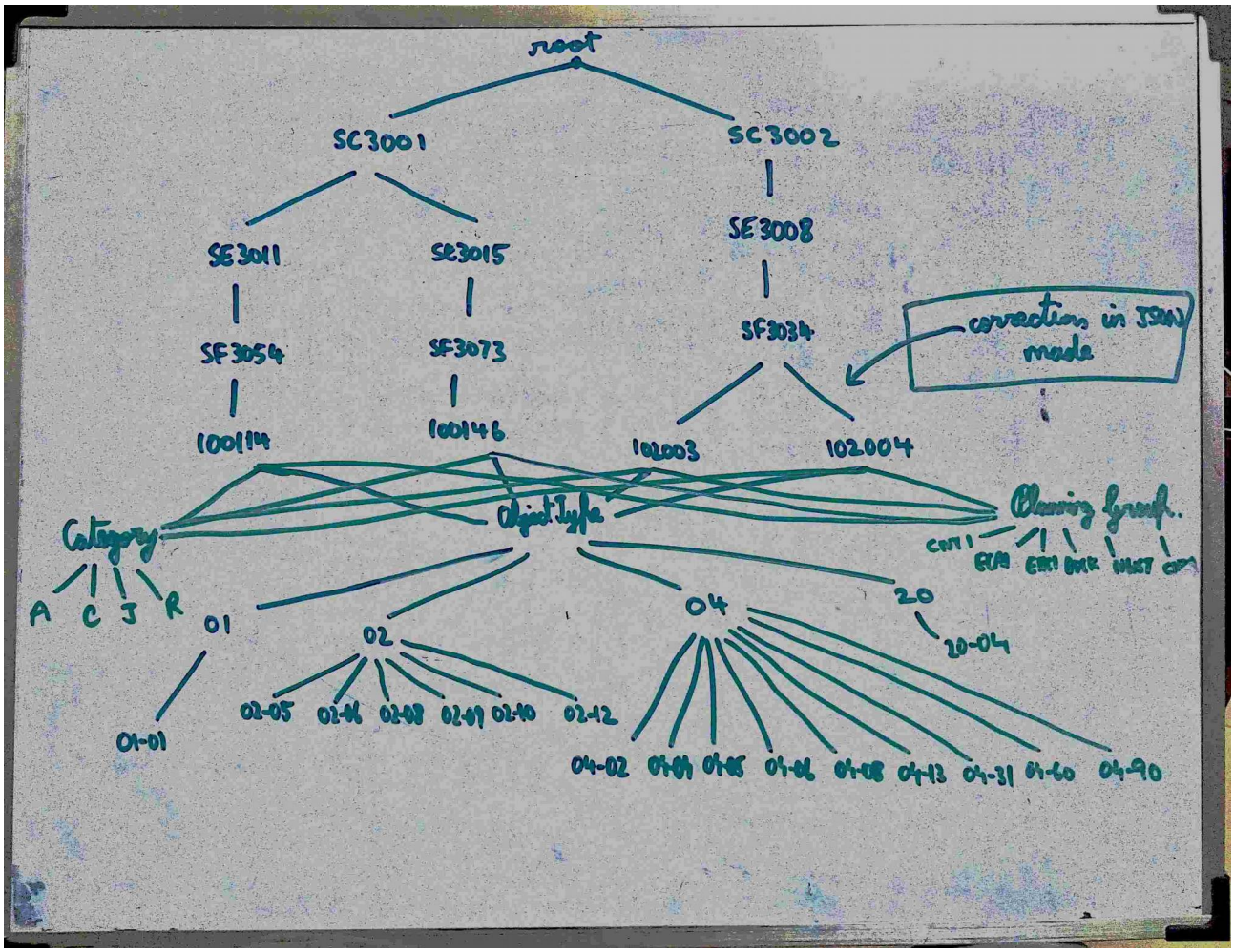
- ***HTML*** – User Interface
- ***CSS*** – Styling the User Interface
- ***JavaScript*** – Graph Path Traversal Algorithm
- ***JSON*** – Storing menu data

CORRECTIONS MADE:

WITHOUT CORRECTION IN JSON FILE:



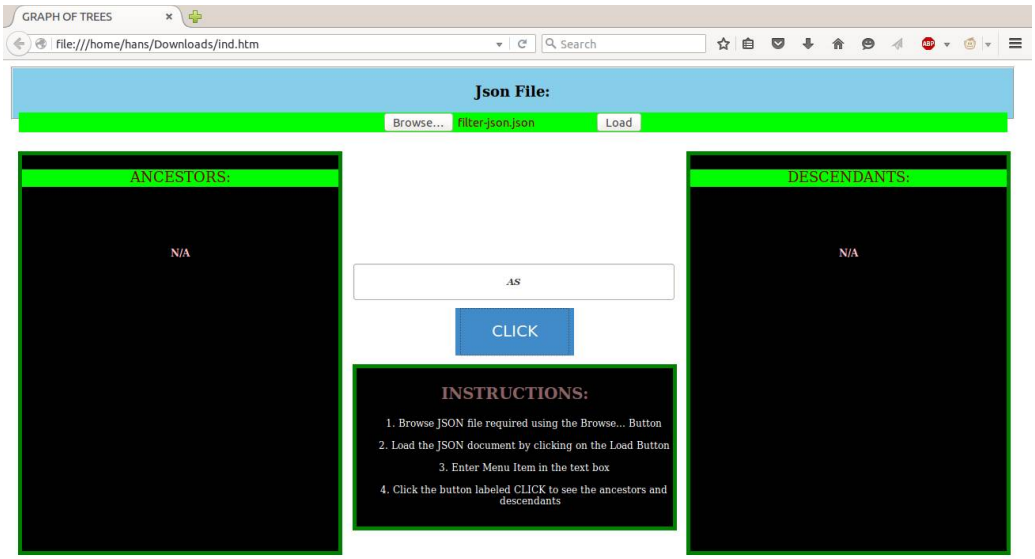
AFTER CORRECTION IN JSON FILE:



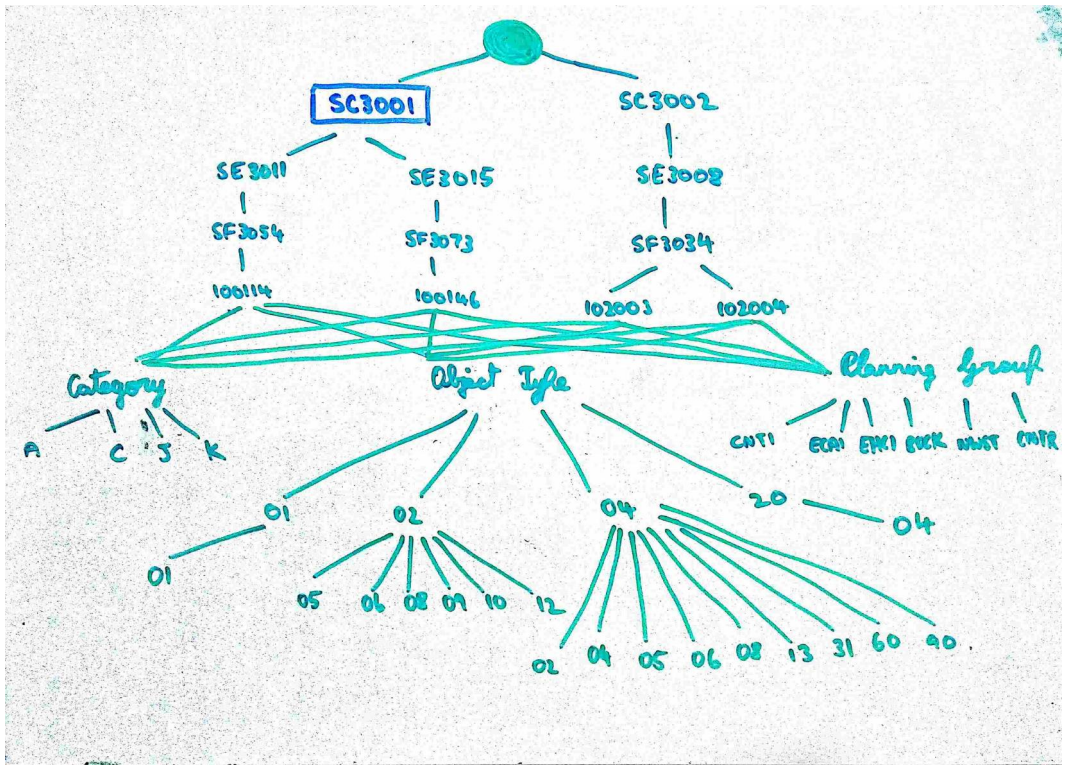
TESTING:

TEST CASE 1: Random Input not present in node list.

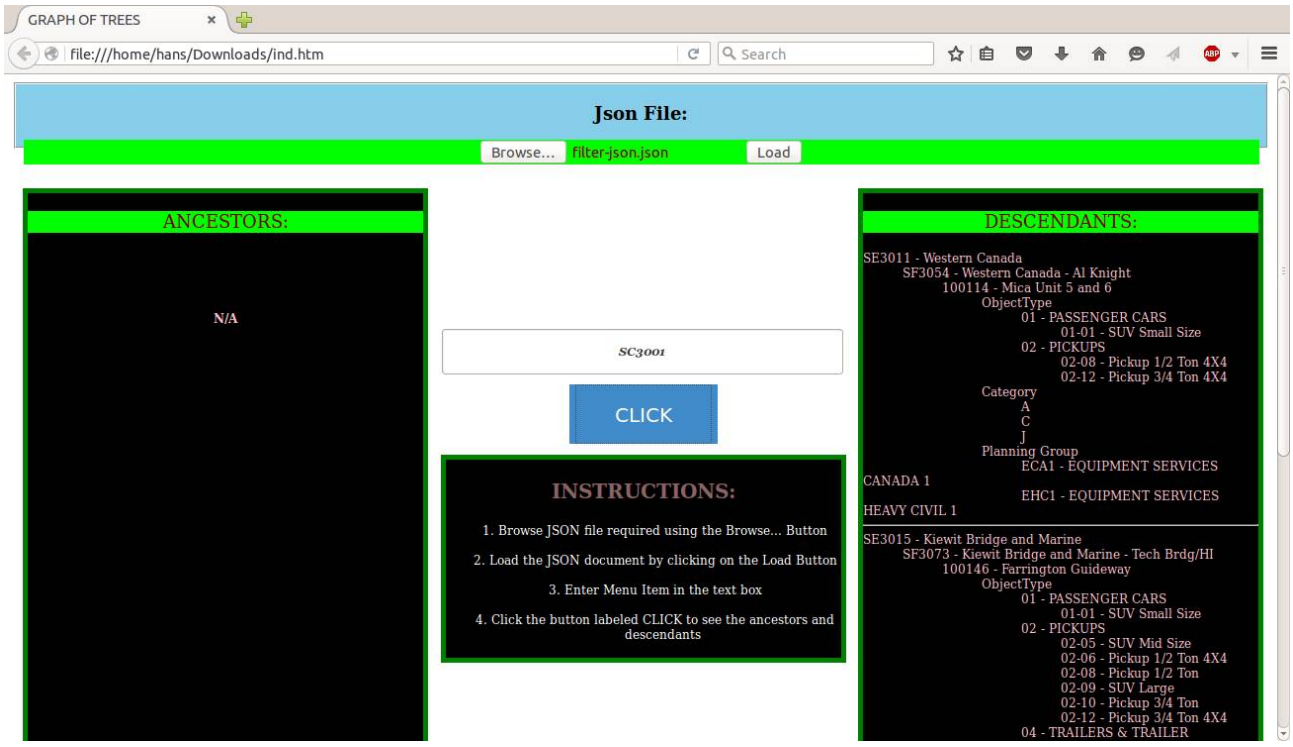
OUTPUT 1: Error Message

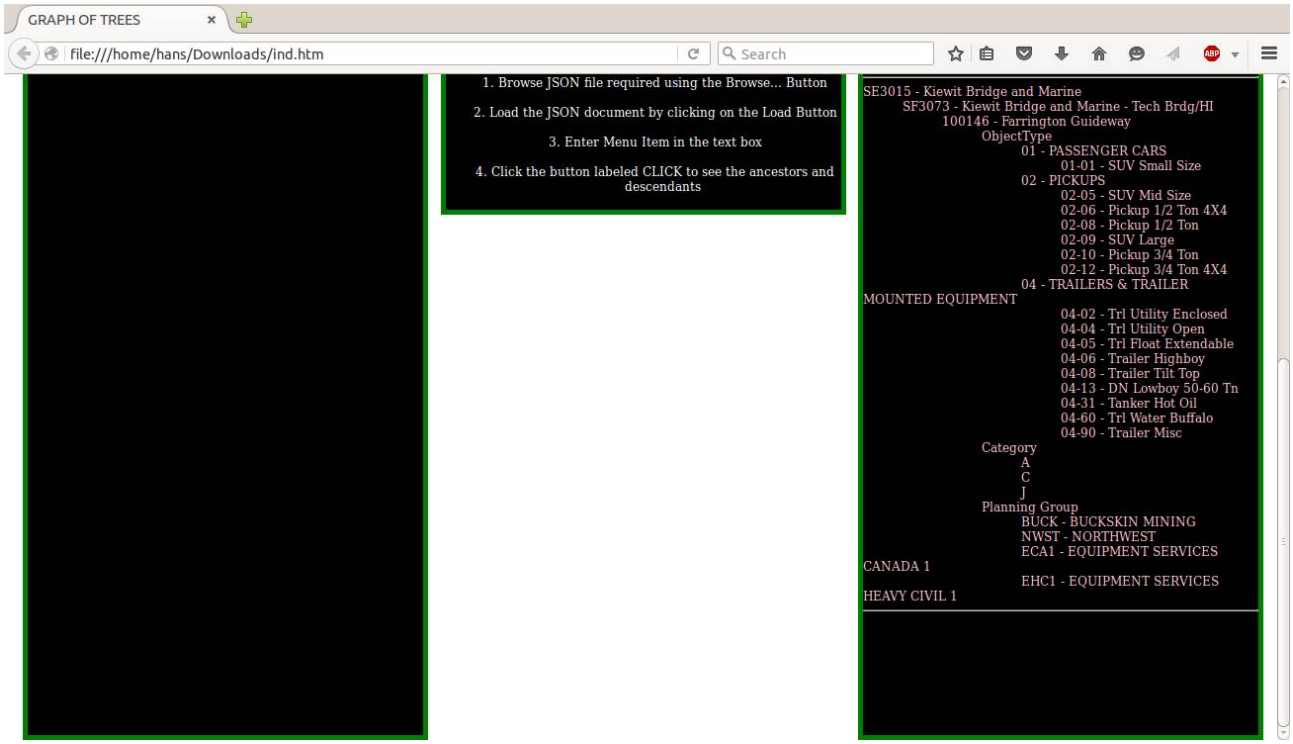


TEST CASE 2:

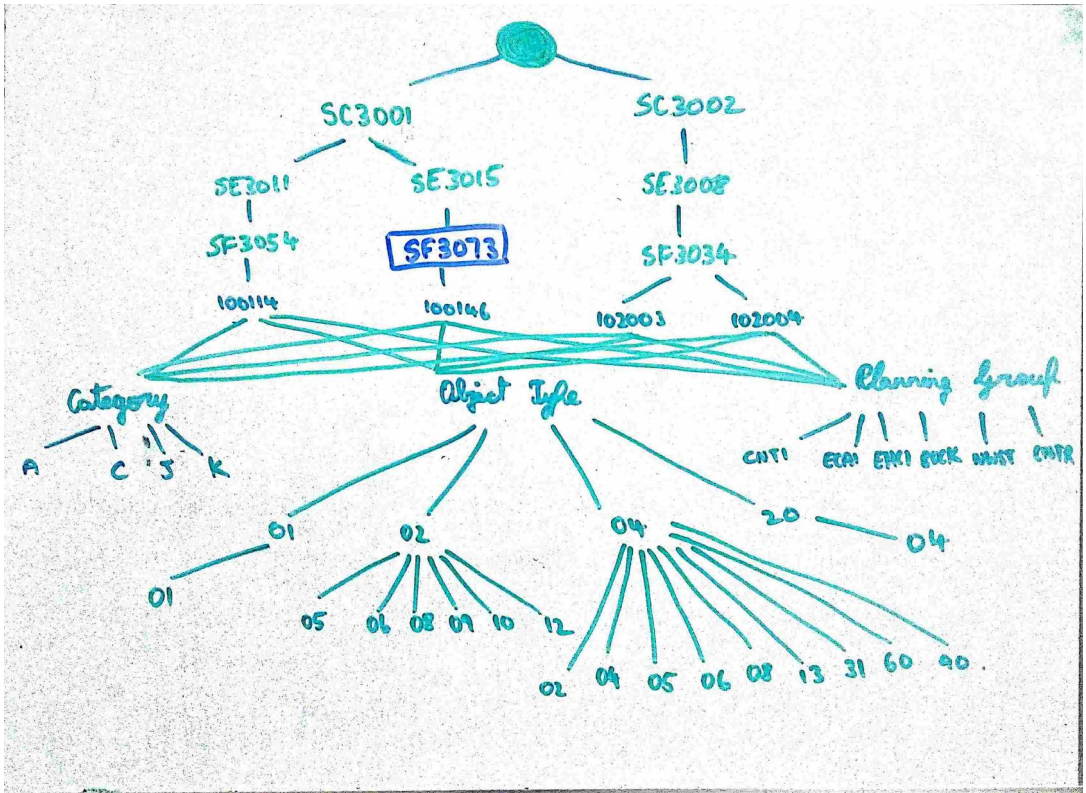


OUTPUT 2:

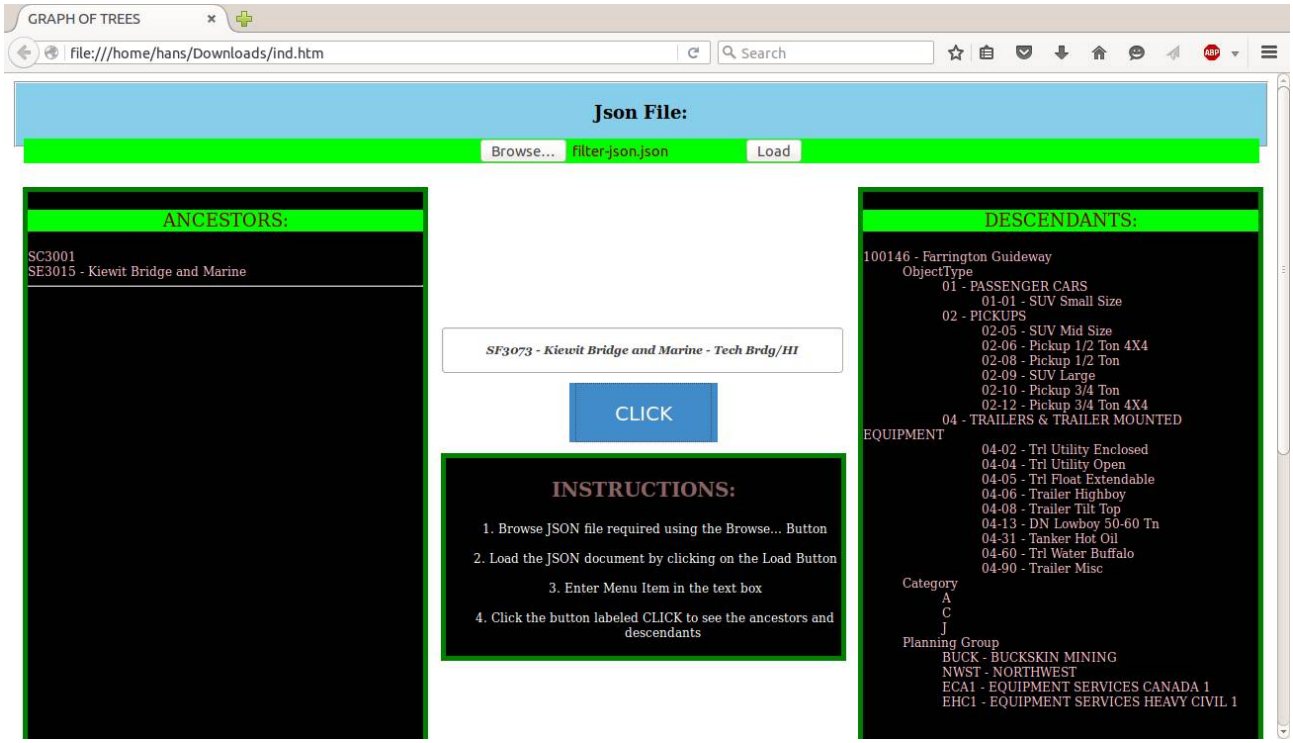



GRAPH OF TREES
file:///home/hans/Downloads/ind.htm
Search
1. Browse JSON file required using the Browse... Button
2. Load the JSON document by clicking on the Load Button
3. Enter Menu Item in the text box
4. Click the button labeled CLICK to see the ancestors and descendants
SE3015 - Kiewit Bridge and Marine
SF3073 - Kiewit Bridge and Marine - Tech Brdg/HI
100146 - Farrington Guideway
ObjectType
01 - PASSENGER CARS
01-01 - SUV Small Size
02 - PICKUPS
02-05 - SUV Mid Size
02-06 - Pickup 1/2 Ton 4X4
02-08 - Pickup 1/2 Ton
02-09 - SUV Large
02-10 - Pickup 3/4 Ton
02-12 - Pickup 3/4 Ton 4X4
04 - TRAILERS & TRAILER MOUNTED EQUIPMENT
04-02 - Trl Utility Enclosed
04-04 - Trl Utility Open
04-05 - Trl Float Extendable
04-06 - Trailer Highboy
04-08 - Trailer Tilt Top
04-13 - DN Lowboy 50-60 Tn
04-31 - Tanker Hot Oil
04-60 - Trl Water Buffalo
04-90 - Trailer Misc
Category
A
C
J
Planning Group
BUCK - BUCKSKIN MINING
NWST - NORTHWEST
ECA1 - EQUIPMENT SERVICES CANADA 1
EHC1 - EQUIPMENT SERVICES HEAVY CIVIL 1

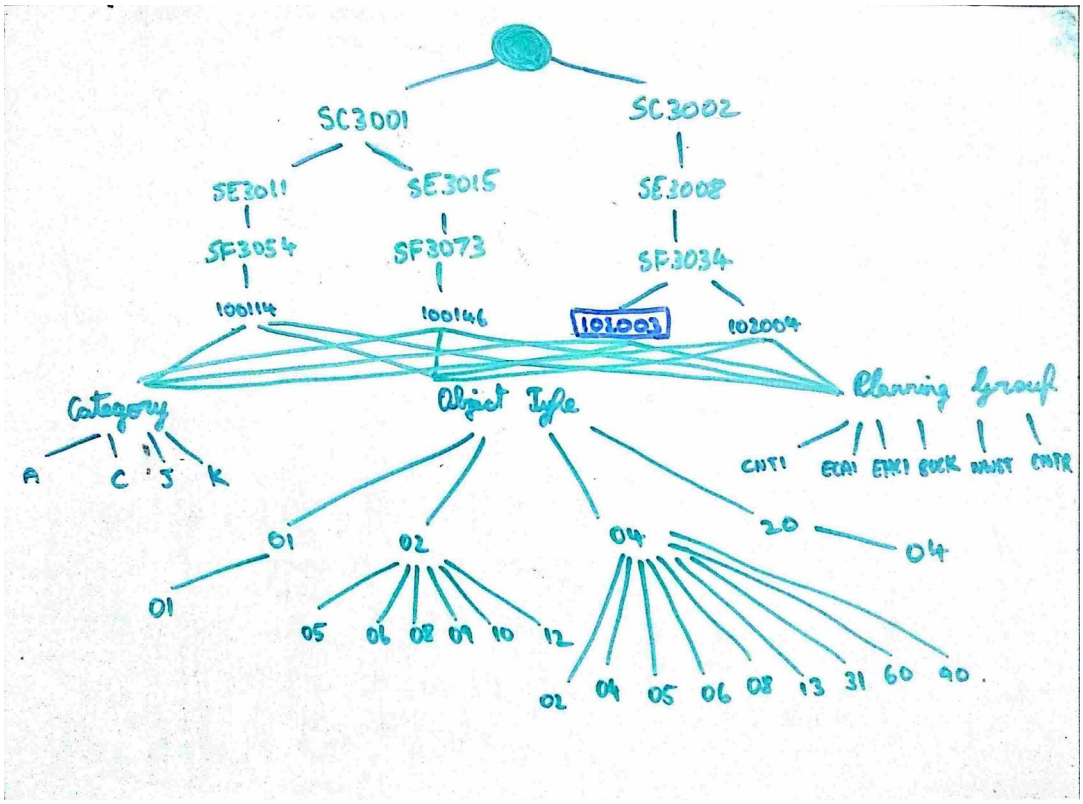
TEST CASE 3:


SC3001
SC3002
SE3011
SE3015
SE3008
SF3054
SF3073
SF3034
100114
100146
102003
102004
Category
Object Type
Planning Group
A
C
J
K
01
02
04
20
01
05
06
08
09
10
12
02
04
05
06
08
13
31
60
90
04

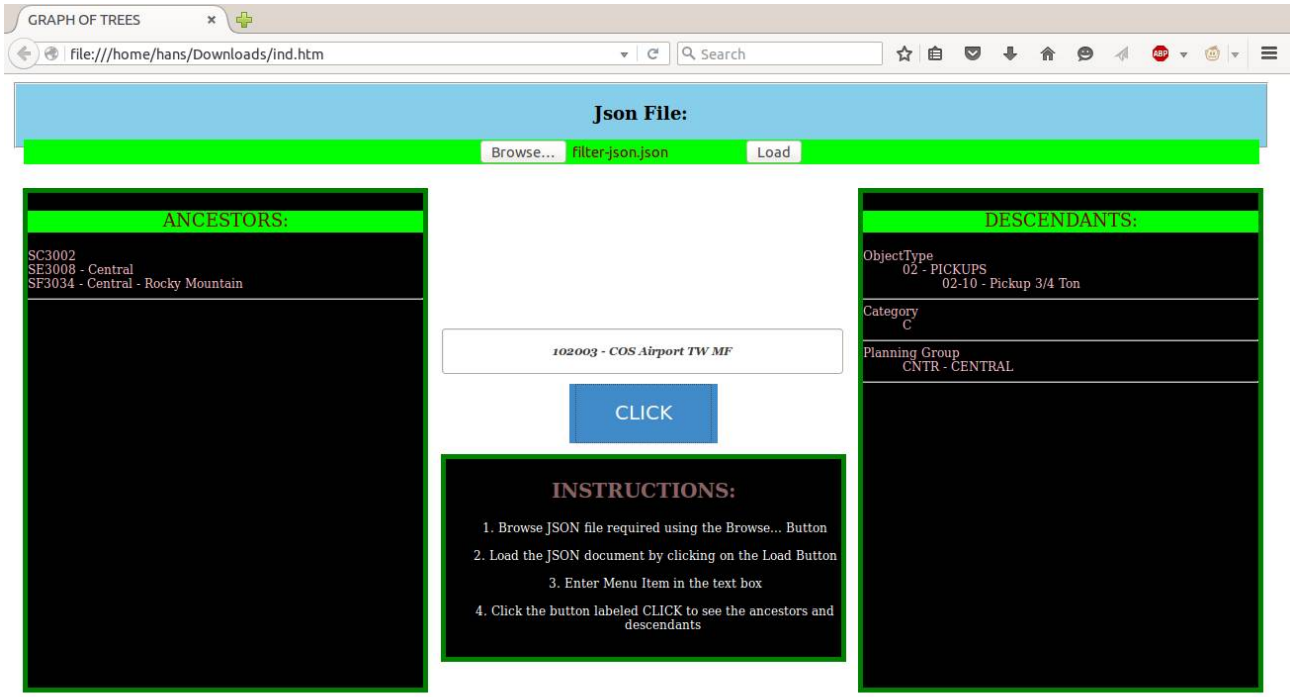
OUTPUT 3:



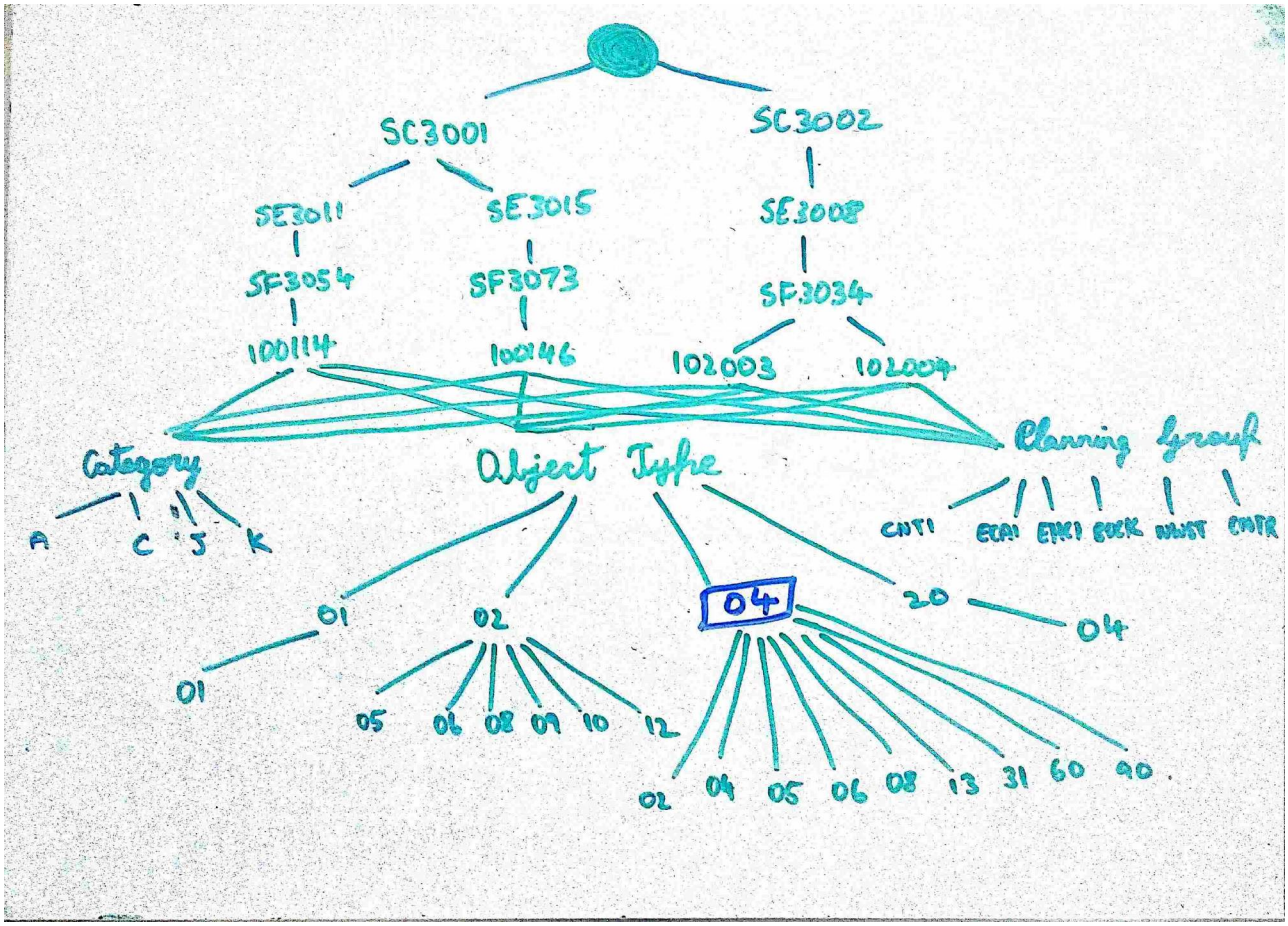
TEST CASE 4:



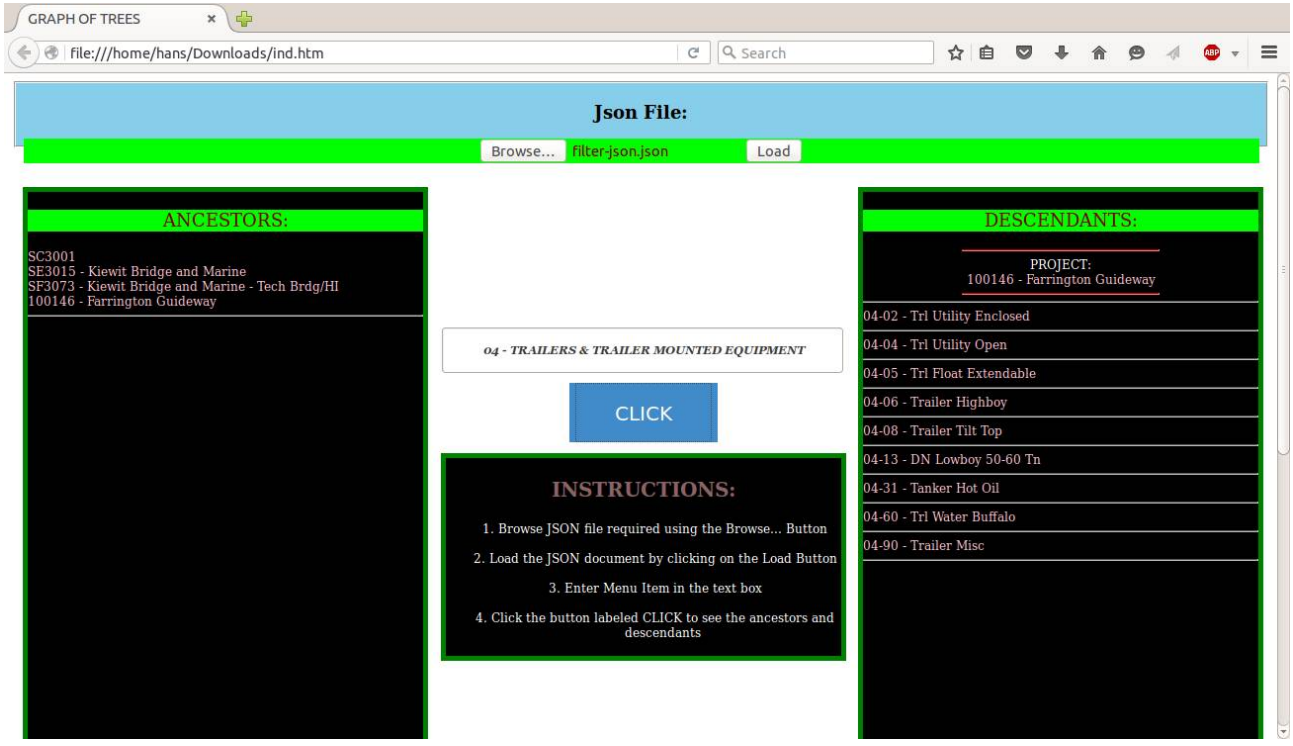
OUTPUT 4:



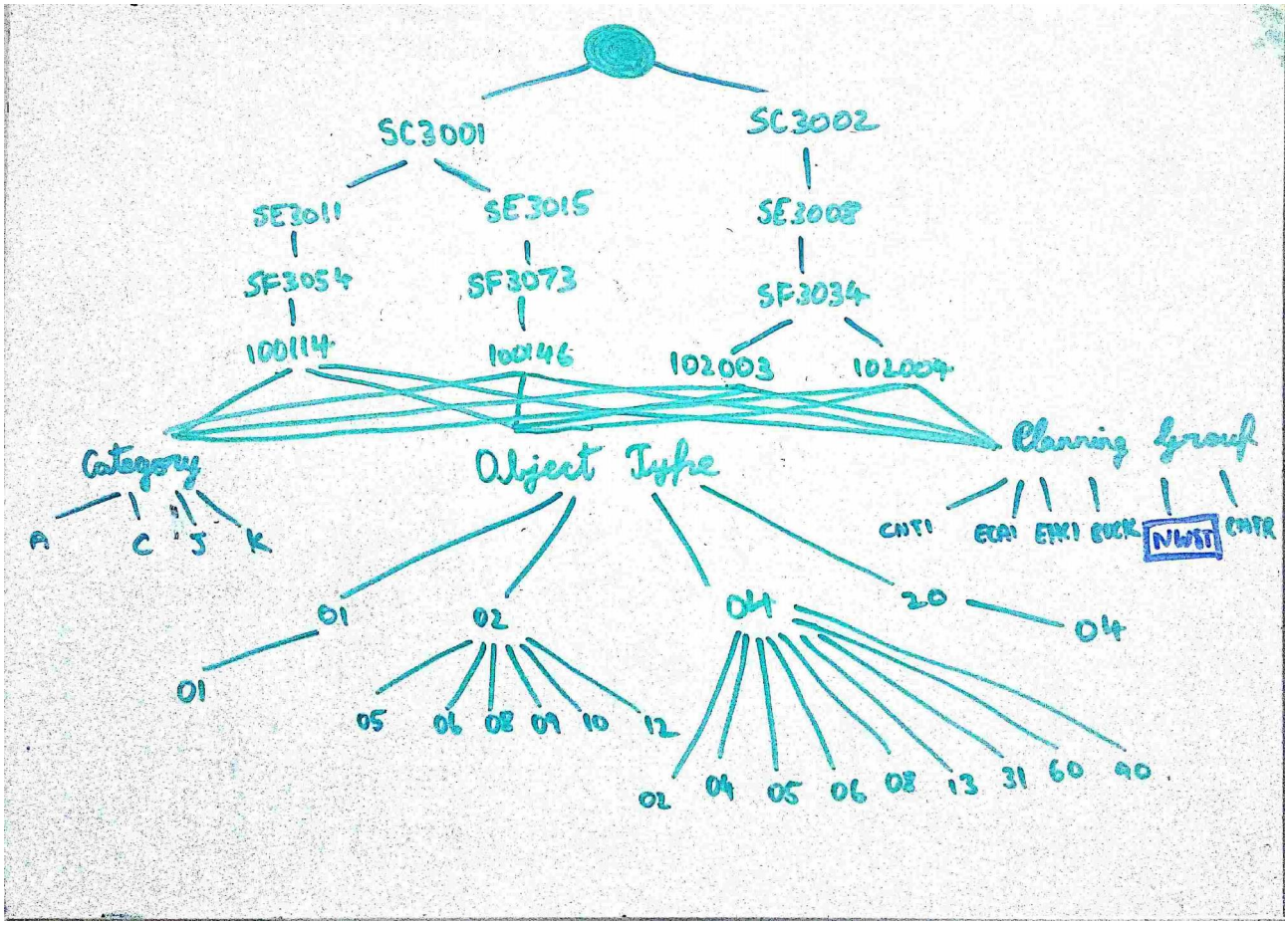
TEST CASE 5:


SC3001
SC3002
SE3011
SE3015
SE3008
SF3054
SF3073
SF3034
100114
100146
102003
102004
Category
Object Type
Planning Group
A
C
J
K
01
02
04
20
01
05
06
08
09
10
12
02
04
05
06
08
13
31
60
90

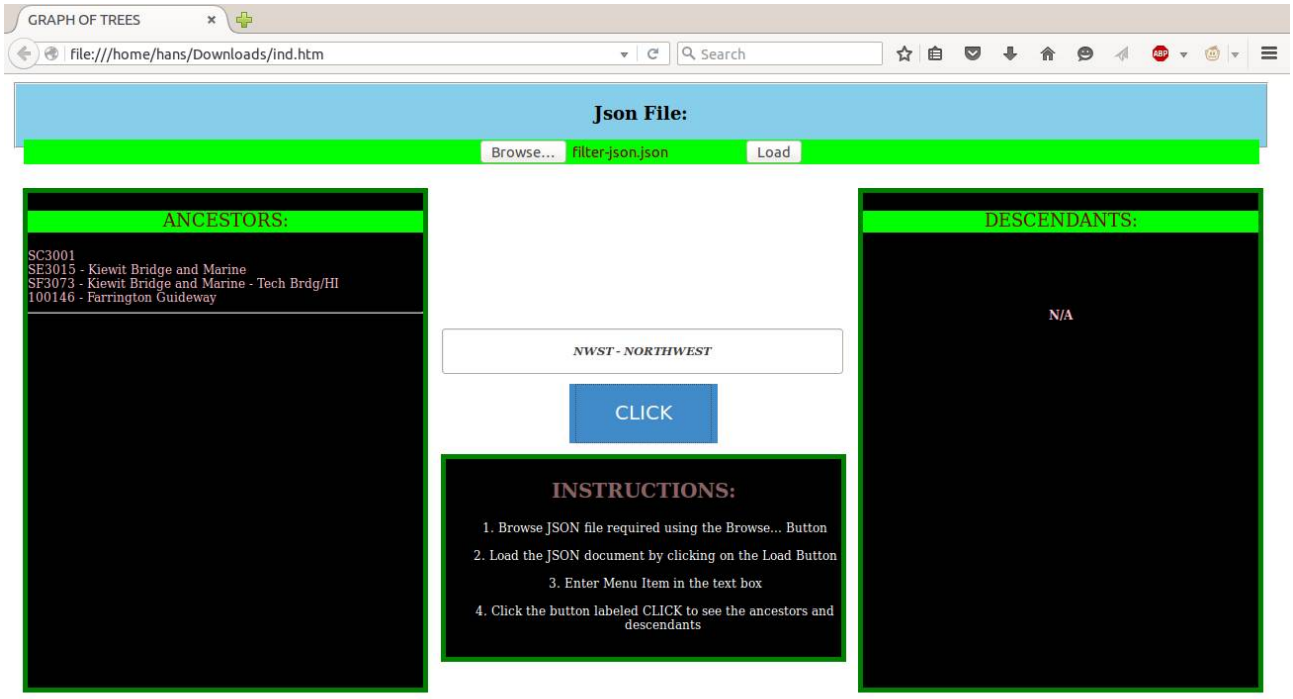
OUTPUT 5:


GRAPH OF TREES
file:///home/hans/Downloads/ind.htm
Search
Json File:
Browse...
filter-json.json
Load
ANCESTORS:
SC3001
SE3015 - Kiewit Bridge and Marine
SF3073 - Kiewit Bridge and Marine - Tech Brdg/HI
100146 - Farrington Guideway
04 - TRAILERS & TRAILER MOUNTED EQUIPMENT
CLICK
INSTRUCTIONS:
1. Browse JSON file required using the Browse... Button
2. Load the JSON document by clicking on the Load Button
3. Enter Menu Item in the text box
4. Click the button labeled CLICK to see the ancestors and descendants
DESCENDANTS:
PROJECT:
100146 - Farrington Guideway
04-02 - Trl Utility Enclosed
04-04 - Trl Utility Open
04-05 - Trl Float Extendable
04-06 - Trailer Highboy
04-08 - Trailer Tilt Top
04-13 - DN Lowboy 50-60 Tn
04-31 - Tanker Hot Oil
04-60 - Trl Water Buffalo
04-90 - Trailer Misc

TEST CASE 6:



OUTPUT 6:

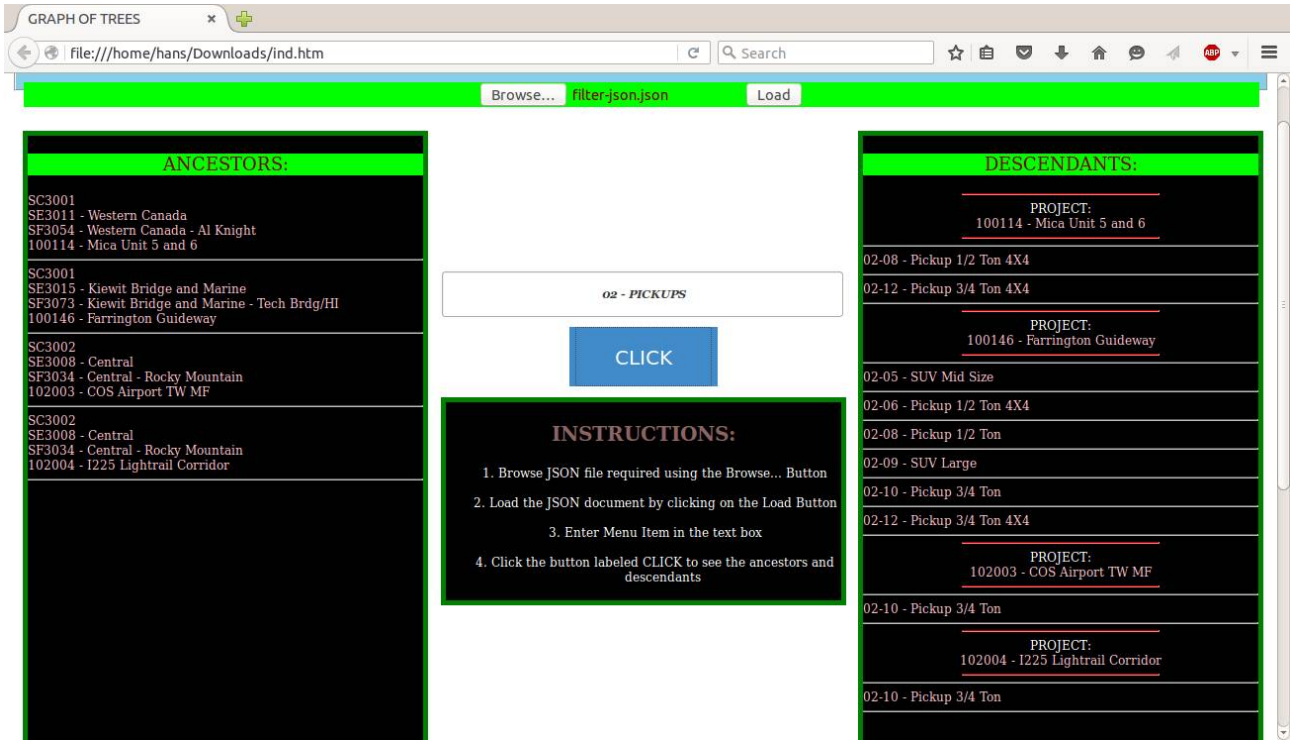


NOTE:

For different objectType elements,

01 - PASSENGER CARS,
02 – PICKUPS,
04 - TRAILERS & TRAILER MOUNTED EQUIPMENT,

the leaf nodes are presented grouped under projects.



REFERENCES:

{1} ***http://www.w3schools.com/***

{2} ***http://stackoverflow.com/***

{3} ***http://www.youtube.com/***