Food On the Way

Version 2.0

April 10th, 2018

Group 17

Harry Fu, Eunice Huang, Jingfan Ai, Yanting Cao, Shengchen Zhou

SE2XB3 Software Engineering Practice and Experience: Binding Theory to Practice

Department of Computing and Software, McMaster University

Table of Content:

Title Page

Revision Page 3

Contribution Page 4

Executive Summary 5

Decomposed Modules in Application 6

Class Description of Main Modules 7-9

View of uses relationship 10

Internal Review 11

Revision Page:

|  |  |  |
| --- | --- | --- |
| Name | Student Number | Role(s) |
| Harry Fu | 400065502 | Client, researcher, designer, programmer, tester |
| Jingfan Ai | 400061906 | Client, researcher, programmer, tester |
| Eunice Huang | 400092172 | Client, recorder, researcher, programmer |
| Shengchen Zhou | 400050783 | Client, designer, programmer, tester |
| Yanting Cao | 400067576 | Client, researcher, programmer, tester |

The roles inside the team is divided into five parts: Client, Researcher, Designer, Programmer, and Tester. The main missions for Client is to raise up the importance of the problem and define the specifications of the expected output. Since the following progress is based on this part. Researcher has to searching relative algorithm that the program may be used, and some similar program which may have similar functions. Meanwhile, researcher should search the basic datasets that the program may need and the similar existing products. Based on the comparison, the improved program could be built. The responsibility for designer is to design the basic algorithm and the overall design (such as the expression of the results, the expression of the input etc.) of the program, which is the guide for programmer. Since the size of the whole program would be big, it is important to distribute different parts of the modules for everyone in the team so that the whole programming efficiency would be raised up. Tester have to debug the whole program, which is also a very crucial work since it could check if there is any problem of the program. To making sure that the program would work perfectly, it is the responsibility for the tester to making sure that there is no problem about the program.

By virtue of submitting this document we electronically sign and date that the work being submitted by all the individuals in the group is their exclusive work as a group and we consent to make available the application developed through SE-2XB3 project, the reports, presentations, and assignments (not including my name and student number) for future teaching purposes.

Contribution Page:

|  |  |  |  |
| --- | --- | --- | --- |
| Name | Role(s) | Contributions | Comments |
| Harry Fu | Client, researcher, designer, programmer, tester | Created the User interface of the project, help to fix and improve the searching algorithm. | The user interface is generated by java window builder which could generate a java interface and convert it to a portable exe file. It could then take an input from users and generate a table for output. |
| Jingfan Ai | Client, researcher, programmer, tester | Design a graph algorithm and an output algorithm. Generate all parts to into a final version of the project. | The graph algorithm does not successfully used in this project since the relevant dataset is missing. The output algorithm runs well. |
| Eunice Huang | Client, recorder, researcher, programmer | Recording relevant documents after each meeting, making project log files, and editing the documentation of the project. (design a sorting algorithm) | The sorting algorithm is not used for the actual project since it does not fully achieve the requirement of the specification. |
| Shengchen Zhou | Client, designer, programmer, tester | Decide the way of reading dataset. Define a class for storing data and later operations in main. Design the sorting algorithm. | The way of storing dataset also decides the way of getting data for calling functions in other part. |
| Yanting Cao | Client, researcher, programmer, tester | Provide stack and union module for the project. Design two classes for searching data. Design the searching algorithm. | Managing the running time of the algorithm. |

Executive Summary:

Nowadays, most people are busy on their jobs. There are too few people have time and chance to go to supermarket every day. Shops are too far away and weather in Canada changes a lot. People can stay at home to order delivery, but the fast foods contains too much fat. Thus, we need vegetables delivered from supermarket. Based on the requirements, the information of the ingredients is important since people have to choose which kind of food they want when they trying to get.

Supermarkets have hundreds of items and each item contains more detailed information such as expiry date. On our project, we tend to provide a platform for people to searching for information of certain food in the grocery stores. The sorting and searching algorithm are being used to manage the dataset and getting result back from the dataset. A user interface is built by java window builder which could generate the function of the project.

Decomposed Modules in Application:

The whole project is decomposed into three main modules: CSVSort, BST, setOutput and the Userinterface. In order to meet the requirement of the project, the project has to generate the dataset in a reasonable and easier way such that it could be simply searched for certain information. The CSVSort is the module which functioned as a sorting algorithm. It sorts the original dataset in a more efficient way so that the sorted dataset could be easily searched. The BST, which is the module contains Binary Search Tree search method, could search through the dataset in short time. The BST module used the subclass which contains three more modules: Bag, Queue, and Stack. These three modules help the BST implemented in an more efficient way such that when coding the BST module, the programmer used less time to write and debug the overall codes. The setOutput module which is used for outputting the result after generating the CSVSort and BST modules. The User interface would help the results which are generated by setOutput become visible and more user friendly. Overall, decomposed the whole project into these three modules and one user interface is a very efficient way to implement the requirement. It helped save a lot of time without writing useless modules for the project.

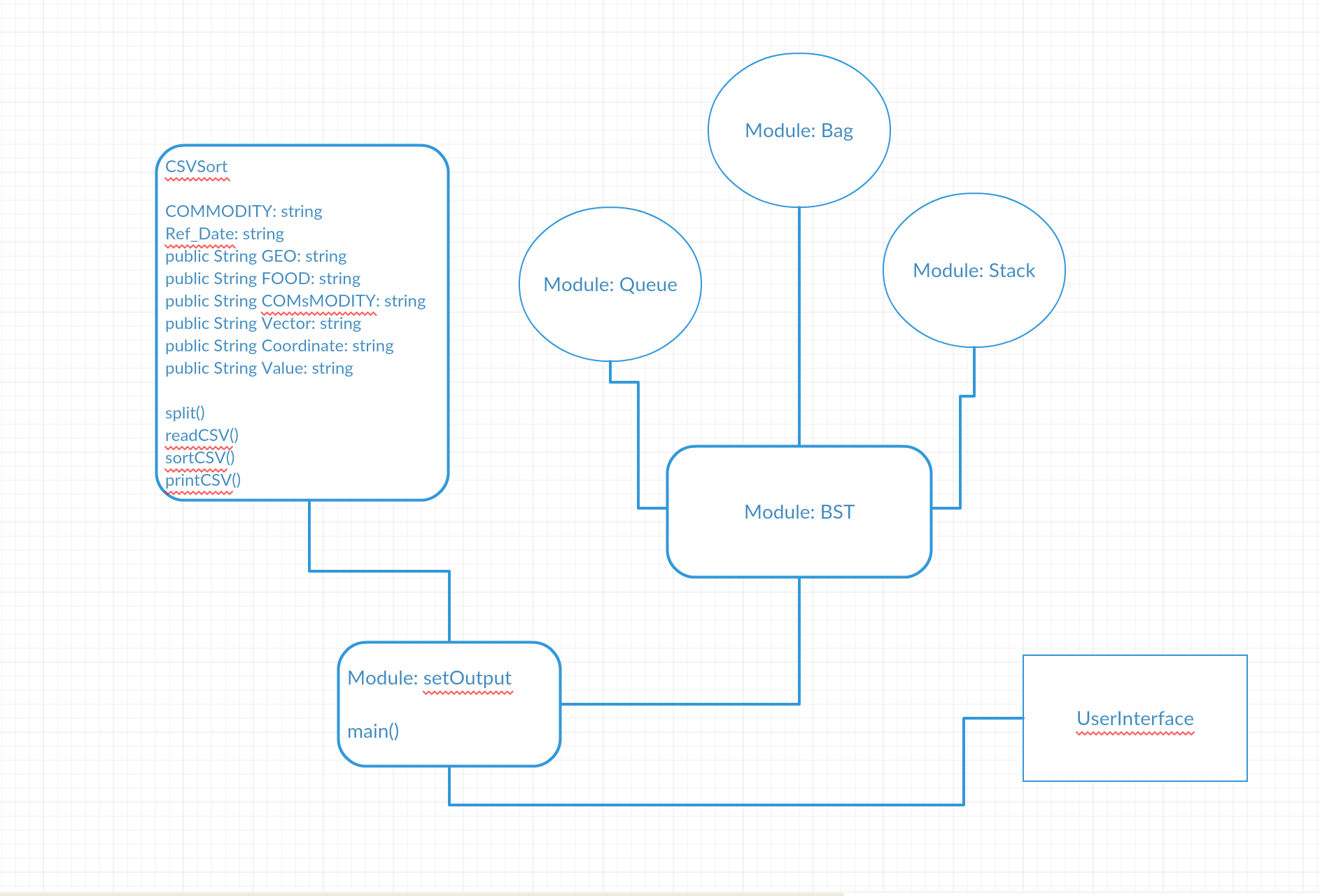


Figure 1: UML Diagram of the decomposed Project

Class Description of Main Modules:

* CSVSort

1. Interface (public entities)

* split(String line): The method would break down the input dataset into strings. And the string would be appended into list after being broke down.
* ReadCSV(String): The method would read the string which is being input.
* SortCSV(List): The method would sort the input list.
* printCSV(String, List): The method would print the string inside the input list.
* gen(): The method would raised a notification to acknowledge that the dataset is being generated.

1. Requirements Achievements: The interface can generate the dataset, which is read the input dataset, stored it as a list and the regenerated the dataset as list. The whole searching operation is based on the sorted dataset, which is generated by this module.

* BST

1. Interface (public entities)

* put (Key key, Value value): This method is putting the key and value into a BST structure. The parameter key is the special symbol of the value and parameter value is the special input.
* get (Key key): This method is getting the value of the key in a BST (Binary Search Tree). The parameter key is the special symbol of the value, and the method would return the value of the key.
* delete (Key key): The method is used for deleting the value in a BST. Parameter key is the special symbol of the value.
* contains (Key key): The method is checking if the input key is in the BST. Parameter key is the special symbol of the value, and the method would return the value of Boolean variable true if the statement is true.
* isEmpty(): The method would check if the node inside the BST is empty or not, and it would return the Boolean variable true if the statement is true.
* size (): The method is checking the size of the BST and it would return an integer which represents the size of the BST.
* min (): The method would return the minimum value inside the BST.
* max (): The method would return the maximum value inside the BST.
* floor (Key key): The method would return the highest key in the BST which is smaller than or equal to the input key. Parameter key is the special symbol of the value and the method would return the value.
* ceiling (Key key): The method would return the smallest key in the BST which is greater than or equal to the input key. Parameter key is the special symbol of the value and the method would return the value.
* rank (Key key): The method would return the rank in the BST, which is the number that is smaller than the input key. Parameter key is the special symbol of the value and the method would return the rank.
* select (int k): The method would return a key inside a BST. Parameter k is the order of the key and it would return the the value.
* deleteMin (): The method would delete the minimum value in the BST.
* deleteMax (): The method deletes the maximum value in the BST.
* size (Key lo, Key hi): The method generates the size between the input lo and the input hi.
* keys (Key lo, Key hi): The method would return the new key list which is between the input parameter lo and the input parameter hi. They are both type Key.
* keys (): The method would return the new key list that is the same as the old one.
* height (): The method would return the height of the root and the result would be an integer value.
* isBST(): The method is checking if the node has a BST. It would return a Boolean variable true if there is BST.
* printLevel(Key key): The method would print the each key in the BST.
* preOrderTraversal(Key key): The method would check if the node has a pre-order traversal. It would return the Boolean variable true if there is pre-order traversal.
* inOrderTraversal(Key key): The method would check if the node has a in-order traversal. It would return the Boolean variable true if there is in-order traversal.
* postOrderTraversal(Key key): The method would generate the post order traversal and print the ouput.

1. Requirements Achievements: Based on the interface listed above, we could see that the BST modules are doing the basic searching algorithm within the dataset provided. Searching algorithm is the main part of the project since the main function of the project is searching data.
2. Implementation (private entities)

* put (Node node, Key key, Value value): The method would put key and value in the correct node of BST. Parameter node is the special place, key is the special symbol of the value. This would return the related node.
* get (Node node, Key key): The method would get the value of the key in a node inside a BST. Parameter key is the special symbol of the value. It would return the value of the key.
* delete(Node node, Key key): The method would find the correct place of the input key and delete the value of the key. Parameter node is the correct node of the key. It would return the value of key.
* size(Node node): The method is checking the size of the node in a BST. It would return the size of the node.
* min (Node node): The method finds the minimum value inside a node. It would return the minimum value.
* max (Node node): The method finds the maximum value inside a BST. It would return the maximum value.
* floor (Node node, Key key): The method would return the highest key in the node which is smaller than or equal to the input key.
* ceiling (Node node, Key key): The method would return the smallest key in the node which is greater than or equal to the key.
* rank (Node node, Key key): The method would return the rank in the node, which is the number of keys smaller than the input key.
* select(Node node, Key key): The method would find the key in a node.
* deleteMin(Node node): The method would delete the minimum value in the BST.
* deleteMax(Node node): The method would delete the maximum value in the BST.
* keys (Node node, List<Key> list, Key lo, Key hi): The method would create the new key list that is just the old one.
* height (Node node): The method would return the height of the node. It would return an integer value.
* isBinaryTree(Node node): The method would check if it is the binary Tree. It would return a Boolean variable true if the input node is a binary Tree.
* isOrdered(Node node, Key min, Key max): The method would check the node if it is in correct order. It would return the Boolean variable true if the isBinaryTree is ordered.
* hasNoDuplicates(Node node): The method would check whether there has duplicates or not. It would return the Boolean variable true if there have no duplicates.
* preOrder(Node node, PrintWriter output): The method would check the node whether there is preOrderTraversal.
* inOrder(Node node, PrintWriter output): The method would check the node whether there is inorder Traversal.
* postPoder(Node node, PrintWrtier output): The method would generate the postOrder function and print the out put after generating.
* setOutput

1. Interface (public entities):

* gettOutput(String food): This is the function which is used to read input. Parameter foo is the name of the input which could be entered by customer or tester.
* getOutput(): This is a function which could return the arraylist. The ouput would return the output arraylist which produced by the gettOutput.
* main(String[] args): The main function is used to generate the output with the function listed above.

1. Requirements Achievements: The interface listed above shows the ability to getting input from the user and getting result back to users.

View of Uses Relationship

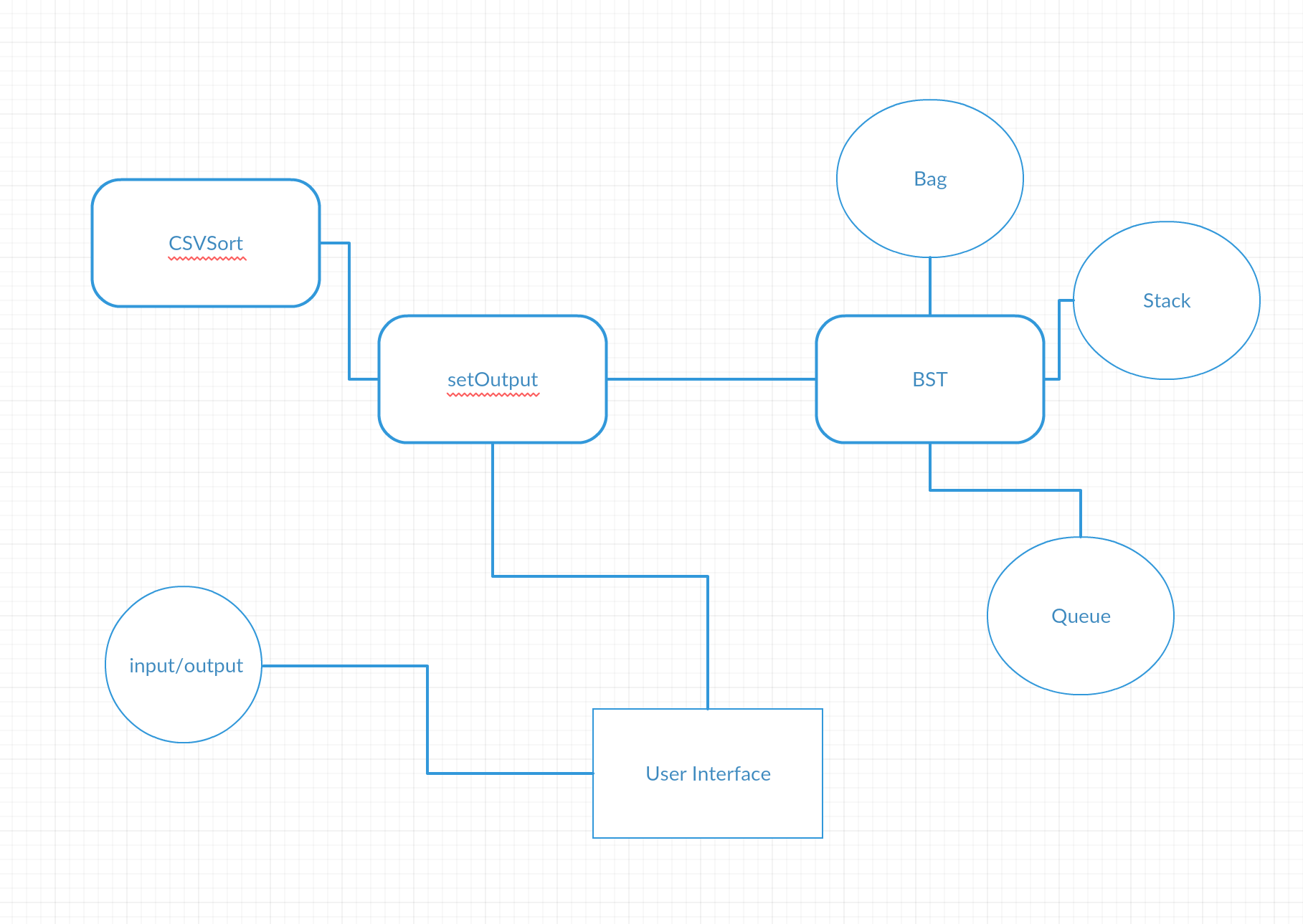


Figure 2: Uses relationship

Internal Review:

During the first demo, there was some errors occur in the BST module. The BST would return the irrelevant information of the input. We soon find that the problem is our searching algorithm run without exceptions. Then we fixed it.

The second demo was not working since there was errors in the user interface. We decided to use java script and HTML to write our interface but then find that it’s hard to learn this new language and we couldn’t fix some bugs in java script reactions. So we decide to do a new interface use java GUI.

The latest version of the user interface is implemented by java window builder and worked perfectly with almost every exception we could imagine.

Overall, the final version of the CSVSort, setOutput, and BST modules are working perfectly.