City University of Hong Kong

Department of Computer Science

Summer Storage Management System for CSSAUG

User Manual

Collaborators:

DONG Jia Jie

LI Xiao Yang

SHA Xin Chen

ZHENG Shang Kun

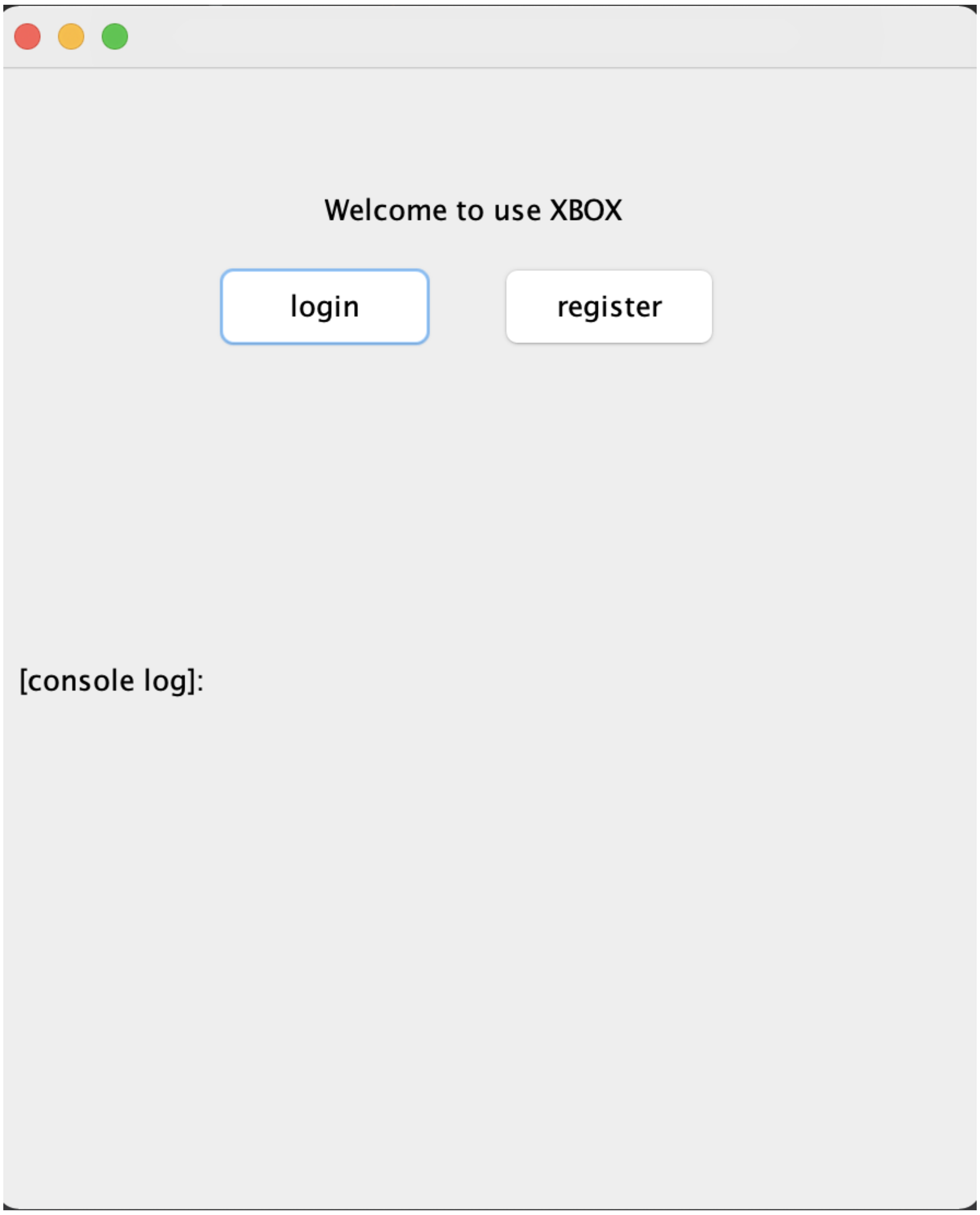
ZHOU Yu

ZHANG Tian Tian

Contents

1 Register & Login

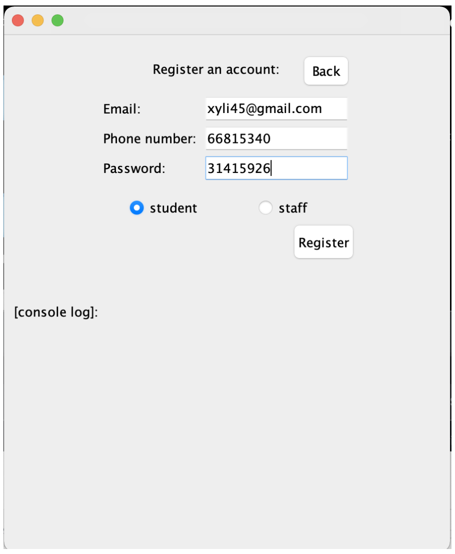
Before users actually executes any commands, they are supposed to register their accounts and login.



```
[console log]:
```

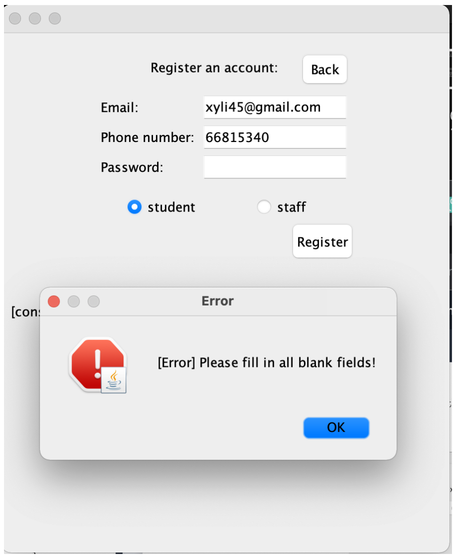
Figure 1: login & register

There are three fields to fill in, which includes email address, phone number and password. It is worth noting that empty input for any of the three fields is illegal.



Email: xyli45@gmail.com
Phone number: 66815340
Password: 31415926
[✓ student] staff

(a) normal flow

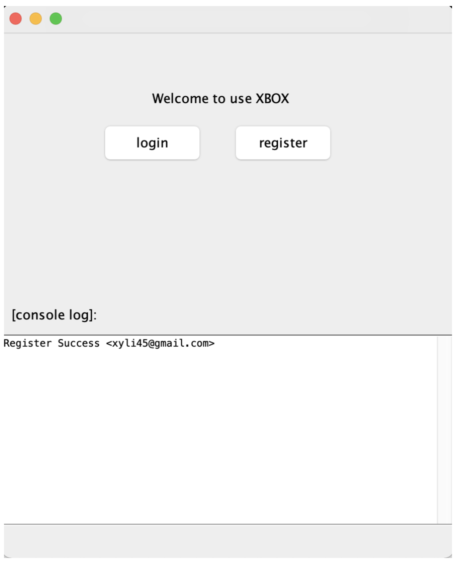


Email: xyli45@gmail.com
Phone number: 66815340
Password:
[✓ student] staff

[Error] Please fill in all blank fields!

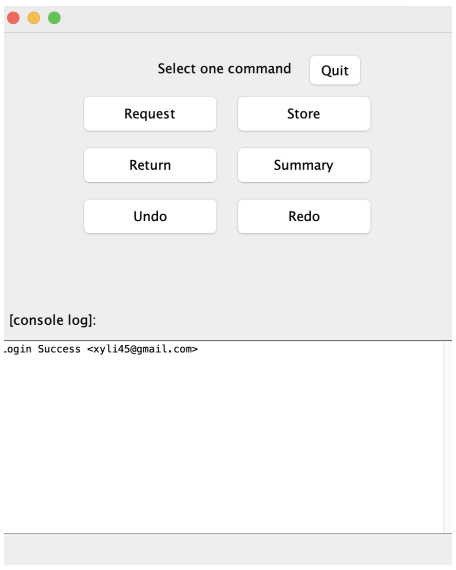
(b) empty field

Once users successfully register or login, they shall see success notifications in the log console below.



```
Register Success <xyli45@gmail.com>
```

(c) register success

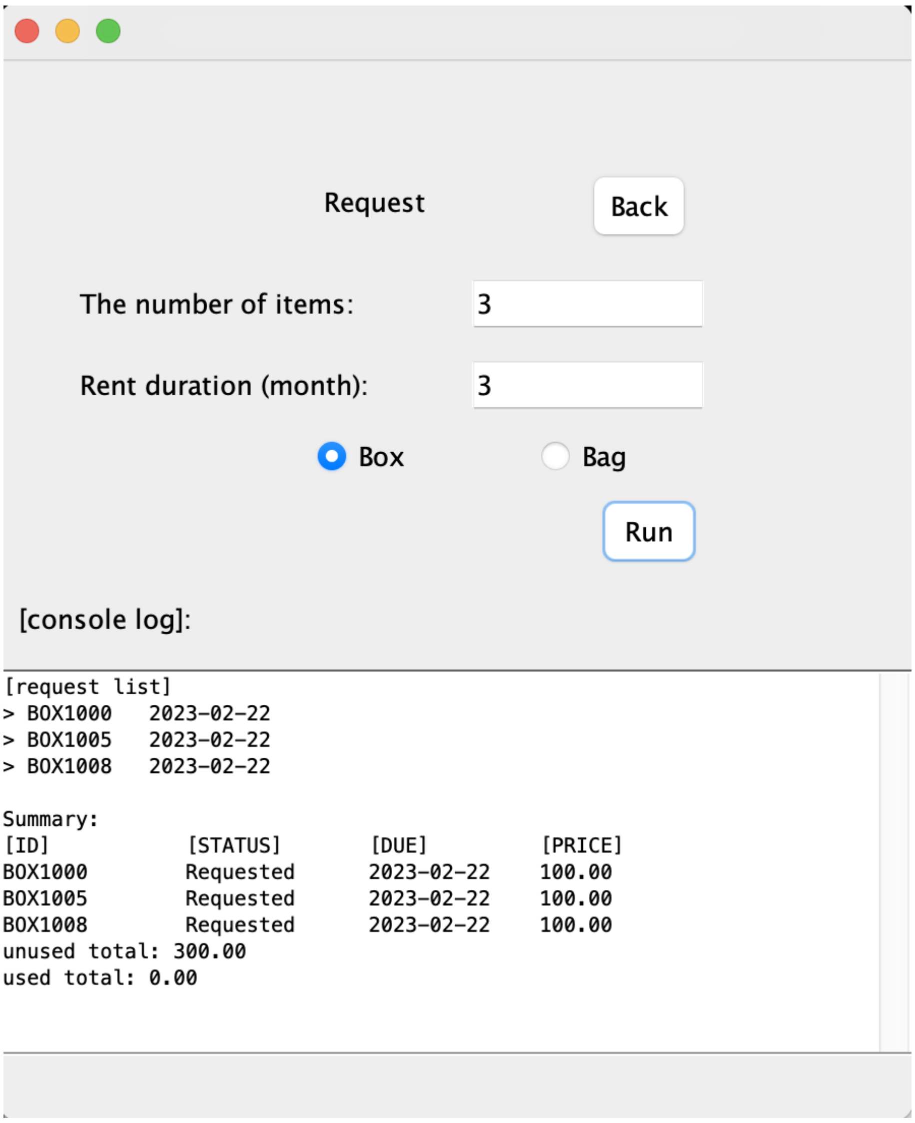


```
.ogin Success <xyli45@gmail.com>
```

(d) login success

2 User Interfaces

Apart from undo and redo functions, four user interfaces including request, store, return and summary are provided. To execute each commands, the client needs to specify input parameters in predefined input format, for example, all segments are separated by blank space delimiters. Therefore, input mistakes such as failing to provide correct delimiters or wrong case sensitive may lead to unexpected program behaviors, which are generally regarded as operation mistakes instead of bugs or program errors.



The number of items: 3

Rent duration (month): 3

[✓ Box] Bag

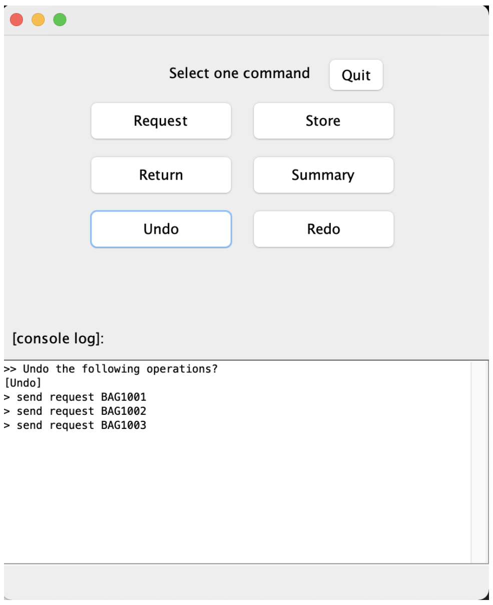
[console log]:

```
[request list]
> BOX1000    2023-02-22
> BOX1005    2023-02-22
> BOX1008    2023-02-22

Summary:
[ID]           [STATUS]       [DUE]         [PRICE]
BOX1000        Requested      2023-02-22    100.00
BOX1005        Requested      2023-02-22    100.00
BOX1008        Requested      2023-02-22    100.00
unused total: 300.00
used total: 0.00
```

Figure 2: runtime demo

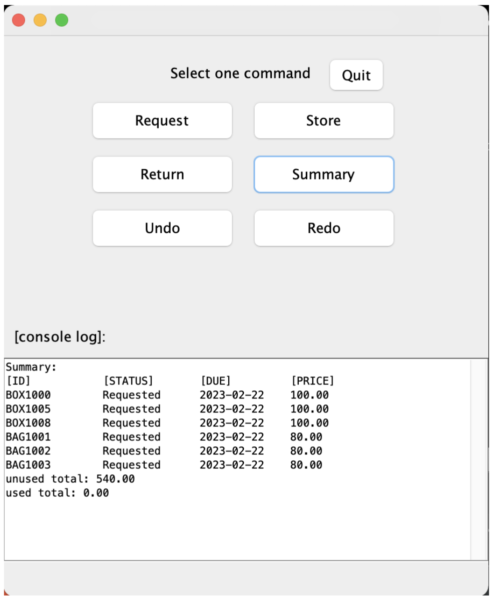
To avoid inconvenience caused by mistakenly specifying parameters, redo and undo interfaces can be utilized to control the work flow flexibly.



```
>> Undo the following operations?
[Undo]
> send request BAG1001
> send request BAG1002
> send request BAG1003
```

Figure 3: Undo

After finishing the tasks, you can click **summary** button to get an overview of the orderings for your reference, for instance, the item status.



```
Summary:
[ID]          [STATUS]      [DUE]         [PRICE]
BOX1000       Requested     2023-02-22    100.00
BOX1005       Requested     2023-02-22    100.00
BOX1008       Requested     2023-02-22    100.00
BAG1001       Requested     2023-02-22    80.00
BAG1002       Requested     2023-02-22    80.00
BAG1003       Requested     2023-02-22    80.00
unused total: 540.00
used total: 0.00
```

Figure 4: summary

Predefined input formats are listed as follows.

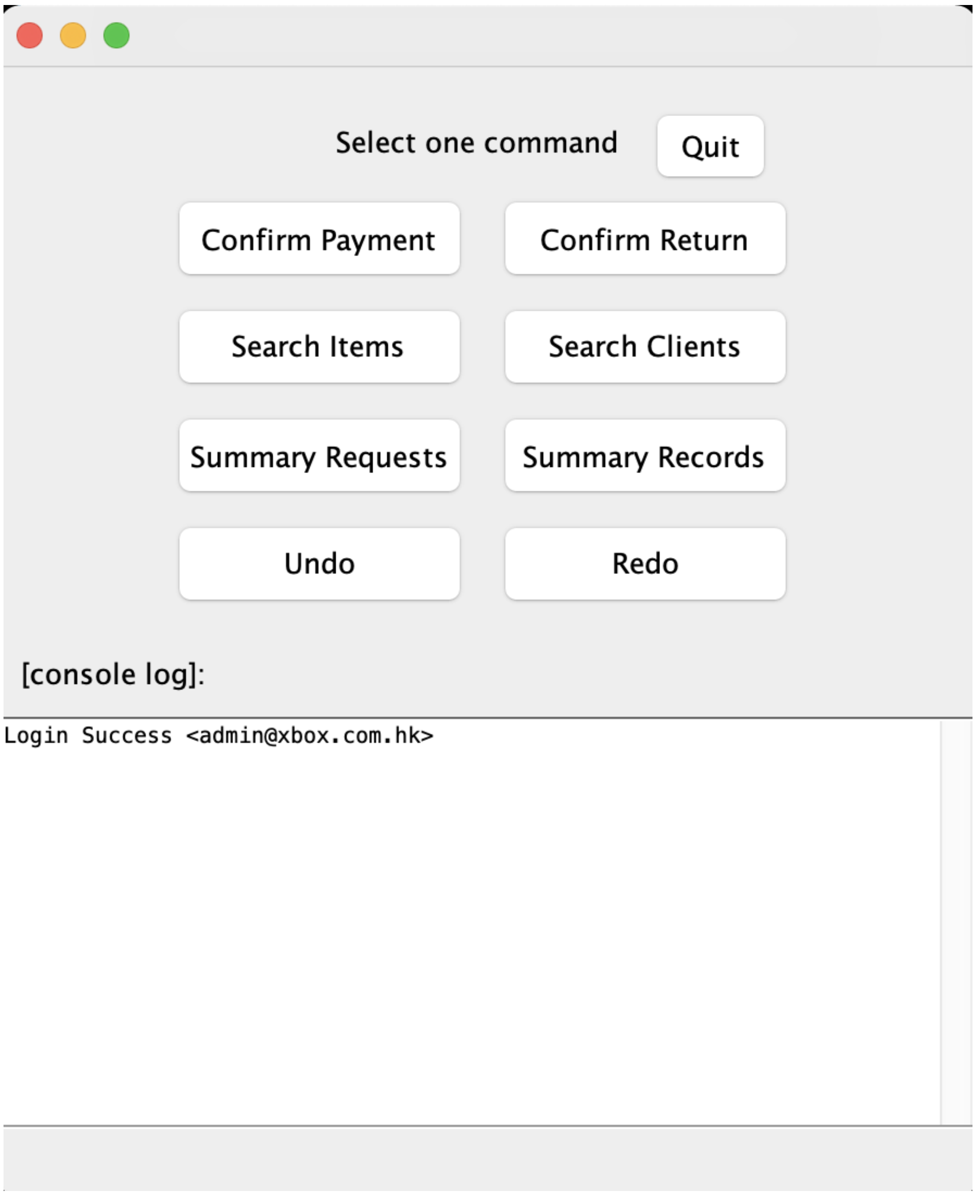
- **Request** [number of items] [rent duration] [type]

- **Store** [item id-1] [item id-2] ... [item id-n]

-**Return** [item id-1] [item id-2] ... [item id-n]

3 Admin Interfaces

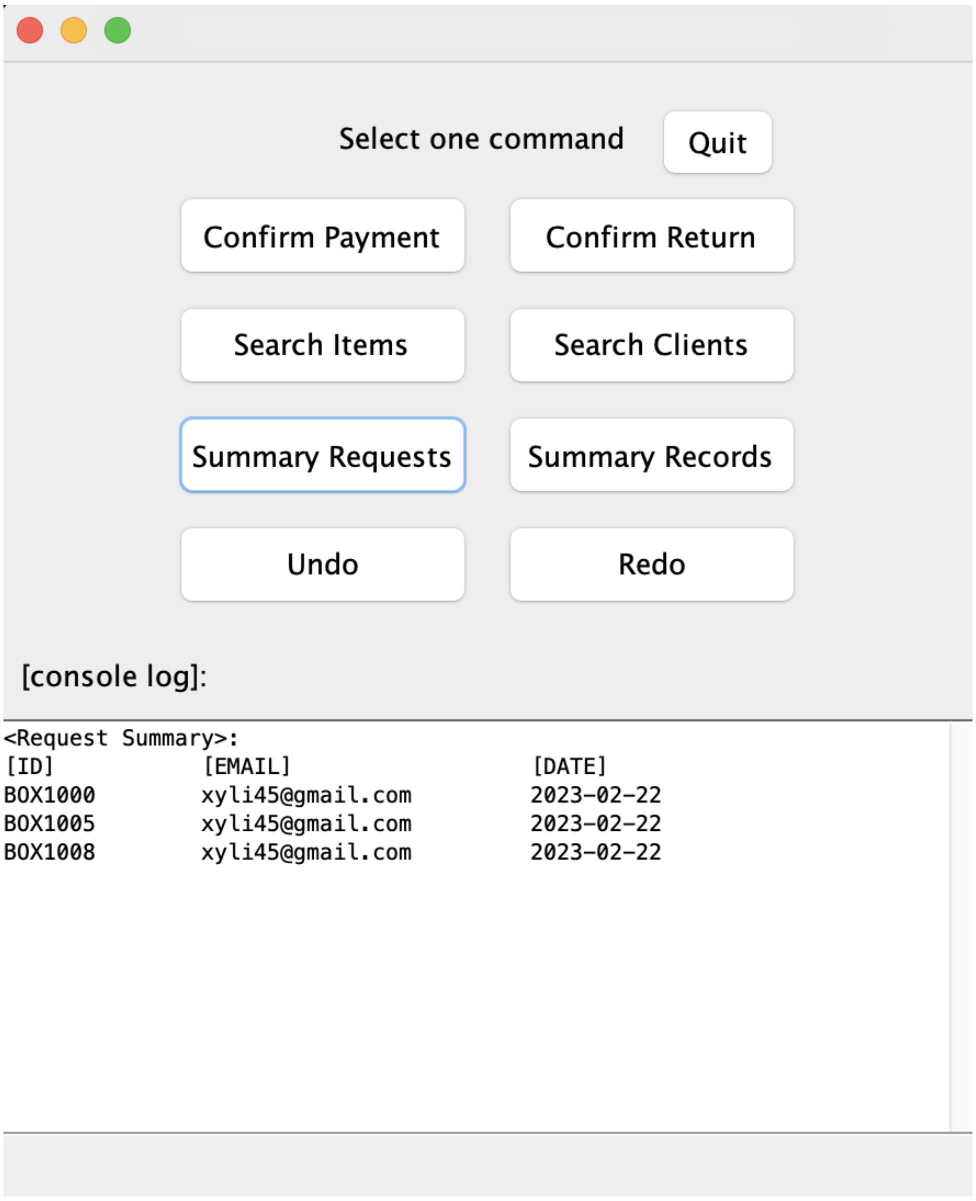
Admin GUI and functions are specially designed for system administrators. Interfaces such as confirming payment, confirming return are provided for them in order to better manage the transaction records.



```
Login Success <admin@xbox.com.hk>
```

Figure 5: customized GUI and interfaces

Here we highlight some new features, for example, search functions and summary functions. Basically you can keep track of information of each and every inventories and clients, which significantly enhance the management efficiency. Similarly, the administrator is able to undo or redo his or her incorrect operation, which is even more essential than that of users.



```
<Request Summary>:
[ID]          [EMAIL]                [DATE]
BOX1000       xyli45@gmail.com       2023-02-22
BOX1005       xyli45@gmail.com       2023-02-22
BOX1008       xyli45@gmail.com       2023-02-22
```

Figure 6: Summary of all orderings

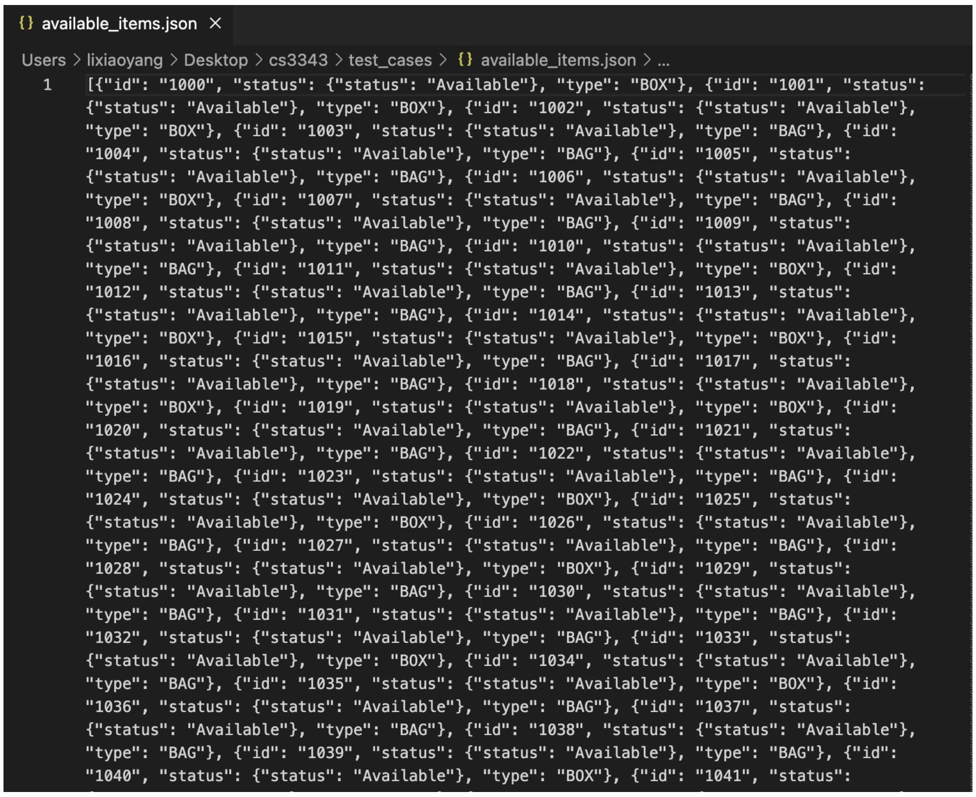
Predefined input formats for admin interfaces are listed as follows.

- **Confirm payment** [email address]

- **Confirm return** [email address] [item id-1] ... [item id-n]

4 Quit

Inspired by microservice architecture, data related to clients, inventories and transactions will be maintained by json file operation. In general, as the program executes, it will firstly read json files to initialize data generated by the last execution. Unsurprisingly, it is when all the data is stored in the json file sets that one quits the program. By doing so, we ensure the consistency of running results between consecutive executions.


{} available_items.json ×
Users > lixiaoyang > Desktop > cs3343 > test_cases > {} available_items.json > ...

```
[{"id": "1000", "status": {"status": "Available"}, "type": "BOX"}, {"id": "1001", "status":
{"status": "Available"}, "type": "BOX"}, {"id": "1002", "status": {"status": "Available"},
"type": "BOX"}, {"id": "1003", "status": {"status": "Available"}, "type": "BAG"}, {"id":
"1004", "status": {"status": "Available"}, "type": "BAG"}, {"id": "1005", "status":
{"status": "Available"}, "type": "BAG"}, {"id": "1006", "status": {"status": "Available"},
"type": "BOX"}, {"id": "1007", "status": {"status": "Available"}, "type": "BAG"}, {"id":
"1008", "status": {"status": "Available"}, "type": "BAG"}, {"id": "1009", "status":
{"status": "Available"}, "type": "BAG"}, {"id": "1010", "status": {"status": "Available"},
"type": "BAG"}, {"id": "1011", "status": {"status": "Available"}, "type": "BOX"}, {"id":
"1012", "status": {"status": "Available"}, "type": "BAG"}, {"id": "1013", "status":
{"status": "Available"}, "type": "BAG"}, {"id": "1014", "status": {"status": "Available"},
"type": "BOX"}, {"id": "1015", "status": {"status": "Available"}, "type": "BOX"}, {"id":
"1016", "status": {"status": "Available"}, "type": "BAG"}, {"id": "1017", "status":
{"status": "Available"}, "type": "BAG"}, {"id": "1018", "status": {"status": "Available"},
"type": "BOX"}, {"id": "1019", "status": {"status": "Available"}, "type": "BOX"}, {"id":
"1020", "status": {"status": "Available"}, "type": "BAG"}, {"id": "1021", "status":
{"status": "Available"}, "type": "BAG"}, {"id": "1022", "status": {"status": "Available"},
"type": "BAG"}, {"id": "1023", "status": {"status": "Available"}, "type": "BAG"}, {"id":
"1024", "status": {"status": "Available"}, "type": "BOX"}, {"id": "1025", "status":
{"status": "Available"}, "type": "BOX"}, {"id": "1026", "status": {"status": "Available"},
"type": "BAG"}, {"id": "1027", "status": {"status": "Available"}, "type": "BAG"}, {"id":
"1028", "status": {"status": "Available"}, "type": "BOX"}, {"id": "1029", "status":
{"status": "Available"}, "type": "BAG"}, {"id": "1030", "status": {"status": "Available"},
"type": "BAG"}, {"id": "1031", "status": {"status": "Available"}, "type": "BAG"}, {"id":
"1032", "status": {"status": "Available"}, "type": "BAG"}, {"id": "1033", "status":
{"status": "Available"}, "type": "BOX"}, {"id": "1034", "status": {"status": "Available"},
"type": "BAG"}, {"id": "1035", "status": {"status": "Available"}, "type": "BOX"}, {"id":
"1036", "status": {"status": "Available"}, "type": "BAG"}, {"id": "1037", "status":
{"status": "Available"}, "type": "BAG"}, {"id": "1038", "status": {"status": "Available"},
"type": "BAG"}, {"id": "1039", "status": {"status": "Available"}, "type": "BAG"}, {"id":
"1040", "status": {"status": "Available"}, "type": "BOX"}, {"id": "1041", "status":
```

Figure 7: json file demo

In this case, service provides are capable of customizing their own use scenario based on real world demand.