```
1   // SPDX-License-Identifier: MIT
2   pragma solidity ^0.8.7;
3
4
5   contract MyAuditEvent {
6
7       struct Audit {
8           string url;
9           string method;
10          string user;
11          uint timestamp;
12          address    account;
13      }
14
15      struct AuditInfo {
16           string name;
17          Audit[] ad_list;
18      }
19
20      mapping(string => AuditInfo) audit_list;
21
22      function put(address account, string[] calldata keys, string[] calldata values) public {
23          string memory name;
24          string memory url;
25          string memory method;
26          string memory user;
27          for (uint i = 0; i < keys.length; i++) {
28              if (keccak256(bytes(keys[i])) == keccak256(bytes('name'))) {
29                  name = values[i];
30              }
31              else if (keccak256(bytes(keys[i])) == keccak256(bytes('url'))) {
32                  url = values[i];
33              }
34              else if (keccak256(bytes(keys[i])) == keccak256(bytes('method'))) {
35                  method = values[i];
36              }
37              else if (keccak256(bytes(keys[i])) == keccak256(bytes('user'))) {
38                  user = values[i];
39              }
40          }
```

```
41
42              AuditInfo storage ad_info = audit_list[name];
43              ad_info.name = name;
44              Audit memory ad = Audit(url, method, user, block.timestamp, account);
45              ad_info.ad_list.push(ad);
46          }
47
48          function get(string[] calldata keys, string[] calldata values) public view returns(AuditInfo memory){
49              // 今回は、取得範囲の指定などはオミット
50              string memory name;
51              for (uint i = 0; i < keys.length; i++) {
52                  if (keccak256(bytes(keys[i])) == keccak256(bytes('name'))) {
53                      name = values[i];
54                  }
55              }
56              return audit_list[name];
57          }
58
59      }
```