

Resources

Emily Herbert

```
1  function login(req) {  
2    function F(resp) {  
3      let u = req.body.username;  
4      let p = req.body.password;  
5      if(resp[u] === p) {  
6        respond('ok');  
7      } else {  
8        respond('error');  
9      }  
10   }  
11   get('passwords.json', F);  
12 }
```

Figure 1. Login

```
1  let c = require('containerless');  
2  
3  function main(req) {  
4    function F(resp) {  
5      let u = req.body.username;  
6      let p = req.body.password;  
7      if (resp[u] === p) {  
8        c.respond('ok');  
9      } else {  
10       c.respond('error');  
11     }  
12   }  
13   c.get('passwords.json', F);  
14 }
```

Figure 2. Login

References

```


1  let c = require('containerless');
2
3  function main(req) {
4      function F(resp) {
5          let u = req.body.username;
6          let p = req.body.password;
7          if (resp[u] === p) {
8              c.respond('ok');
9          } else {
10             c.respond('error');
11         }
12     }
13     c.get('passwords.json', F);
14 }
15
16 c.listen(main);

```


```

1  let c = require('containerless');
2  let t = require('containerless/tracing');
3
4  function main(req) {
5      let [_req] = t.popArgs();
6      function F(resp) {
7          let [_resp] = t.popArgs();
8          let _clos = t.popClosure();
9          t.let('req', t.getClos(_clos, 'req'));
10         let u = req.body.username;
11         t.let('u', t.get(t.get(t.id('req'), 'body'), 'username'));
12         let p = req.body.password;
13         t.let('p', t.get(t.get(t.id('req'), 'body'), 'password'));
14         t.if(t.eq(t.vget(_resp, t.id('u')), t.id('p')));
15         if (resp[u] === p) {
16             t.ifTrue();
17             t.pushArgs(t.str('ok'));
18             c.respond('ok');
19             t.popResult();
20         } else {
21             t.ifFalse();
22             t.pushArgs(t.str('error'));
23             c.respond('error');
24             t.popResult();
25         }
26         t.exitIf();
27         t.exitFunction(t.undefined);
28     }
29     t.let('F', t.closure({ 'req': _req }));
30     t.pushArgs([t.str('passwords.json'), t.id('F')]);
31     c.get('passwords.json', F);
32     t.popResult();
33 }
34
35 c.listen(main);

```


if (resp[u] === p) {respond('ok'); } **else** {  }

(a) The IR program produced when the username and password combination is correct.

if (resp[u] === p) {  } **else** { respond('error'); }

(b) The IR program produced when the username and password combination is incorrect.

```

1  if(resp[u] === p) {
2      respond('ok');
3  } else {
4      
5  }

```

Resources

```
1  if(resp[u] === p) {
2    respond('ok');
3  } else {
4    respond('error');
5  }

1  let u = req.body.username;
2  let p = req.body.password;

1  event('listen', [], closure(), callback (clos, req) {
2    event('get', ['passwords.json'], closure(&req),
3      callback(clos, resp) {
4        let req = *(clos.req);
5        // body
6      });
7  });

1  event('listen', [], closure(), callback (clos, req) {
2    event('get', ['passwords.json'],
3      closure(&req), callback(clos, resp) {
4        let req = *(clos.req);
5        let u = req.body.username;
6        let p = req.body.password;
7        if (resp[u] === p) {
8          respond('ok');
9        } else {
10         respond('error');
11       }
12     });
13  });
```