The NodeJS app should include three pages:

1. Single window with an input panel (nothing else). Maybe a submit button.
2. An error page providing instructions to contact someone
3. A success window that displays the final payload, along with instructions on how to use

Workflow:

1. User provides auth token to the NodeJS app
   1. If there is an error in authentication, display error page
   2. If successful, move to step 2
2. Vault returns payload with session token (client\_token)
   1. Parse client\_token
   2. If there is no client\_token, display error page
   3. If successful move to step 3
3. NodeJS app uses session token to retrieve secret
4. Vault returns payload – which includes secret
5. NodeJS app provides display of the payload (nicely formatted)

We use a GitHub personal token to login to Vault. The objective is to have Vault authenticate the user and confirm the identity. The Vault server is located at:

http://35.196.17.47:8200

The token value for the test is the following:

65fa29d416909e9867c794ae6792999f2d251aaa

Here is a simple example of the curl command that allows for the call-in.

|  |
| --- |
| curl \  --request POST \  --data '{"token": "65fa29d416909e9867c794ae6792999f2d251aaa"}' \  <http://35.196.17.47:8200/v1/auth/github/login> |

This is a sample of the expected payload returned by the Vault server. The importance of this payload is the client\_token as it allows for the next operation.

{

"request\_id": "b7c7b5c5-e7be-29f7-e1ab-fabf3ebf5daf",

"lease\_id": "",

"renewable": false,

"lease\_duration": 0,

"data": null,

"wrap\_info": null,

"warnings": null,

"auth": {

"client\_token": "s.j21DuIQZGnCWTbXwu2uD2kGq",

"accessor": "QWLtt05sTFB8sdSvrgrujLh7",

"policies": [

"default",

"dev-policy"

],

"token\_policies": [

"default"

],

"identity\_policies": [

"dev-policy"

],

"metadata": {

"org": "interrupt-software",

"username": "gcdata-admin"

},

"lease\_duration": 2764800,

"renewable": true,

"entity\_id": "8dea795e-7f86-8dc9-3e3b-2c19ae49a833",

"token\_type": "service",

"orphan": true

}

}

Given the client\_token in the operation, the application is now able to request the secret located in the vault. The endpoint for that secret is different than above. The following is a sample curl command to retrieve the secret:

|  |
| --- |
| curl \  --header "X-Vault-Token: s.j21DuIQZGnCWTbXwu2uD2kGq" \  <http://35.196.17.47:8200/v1/secret/nodejs-app> |

This is the final payload to be displayed on page two of the NodeJS application.

{

"request\_id": "b379471f-a2ef-03a0-0b4f-949745e62225",

"lease\_id": "",

"renewable": false,

"lease\_duration": 2764800,

"data": {

"secret": "OrgvU84XLJ6vnPrT50xyao05"

},

"wrap\_info": null,

"warnings": null,

"auth": null

}