Just another cybersecurity case study

Jaden Furtado

Foreword:

This hack was done by me because a friend of mine, challenged me to break into this website, which I accepted. This hack has been done for the sole purpose of having fun! No monetary gains from this!

I guess I will just keep putting out case studies as and when I take down a website. But this time, it was just sad. LOL

I have tried to explain the exploit as best as I can. I have assumed that the reader is familiar with web-development and basic computer science. If not, you can always google the terms I have used or you can ask me to explain parts that you have not understood.

I have not disclosed the company name or any personal/business data. The website in its entirety belongs to said company. I do not own the company or the website.

I have not been hired by said company and do not work for them.

All the images(screenshots) you see here, of the website, are real. They were taken by me at the time of doing the hack. I have blurred the name of the company to protect their privacy until they can fix the problems I have flagged.

I do not intend to break the law in any way. This has been done by me purely for fun and to raise awareness of cybersecurity.

Jaden Furtado

NOTE: This is a step-by-step documentation of how to hack the website. I have shown how a hacker can use this site for malicious purposes, in this document. I respect the privacy of the agencies involved and I don't intend any breach of law or harm in any way. That being said, I feel the need to document this to get the agencies to take me seriously and fix the underlying issues.

Note: This document is to be used only for the purpose of documentation and fixing this site. This document does not stand as evidence and cannot be used as evidence in any court of law in India or any other country.

Disclaimer: Hacking is a punishable offense in India with imprisonment up to 3 years, or with fine up to two lakh rupees, or with both. Chapter IX Section 43 of IT act, 2000 prescribes a penalty for the damage to computer or computer system. It is a common thing which happens whenever a computer system is hacked. Black hats damage the system that they hack and steal the information. This enumerative provision includes a lot of activities. I do not encourage hacking in any way and am doing this solely with an educational objective. I am doing this in the best interest of the company.

---

glossary

ASP.NET:

ASP.**NET** is an open source web **framework**, created by Microsoft, for building modern web apps and services with . **NET**. **ASP**.**NET** is cross platform and runs on Linux, Windows, macOS, and Docker.

Session:

A **session** can be defined as a **server**-side storage of information that is desired to persist throughout the user's interaction with the web site or web application. ... This **session** id is passed to the web **server** every time the browser makes an HTTP request (ie a page link or AJAX request).

SQL:

SQL stands for Structured Query Language. It is designed for managing data in a relational database management system (RDBMS). It is pronounced as **S-Q-L** or sometime See-Qwell. **SQL** is a database language, it is used for database creation, deletion, fetching rows, and modifying rows, etc.

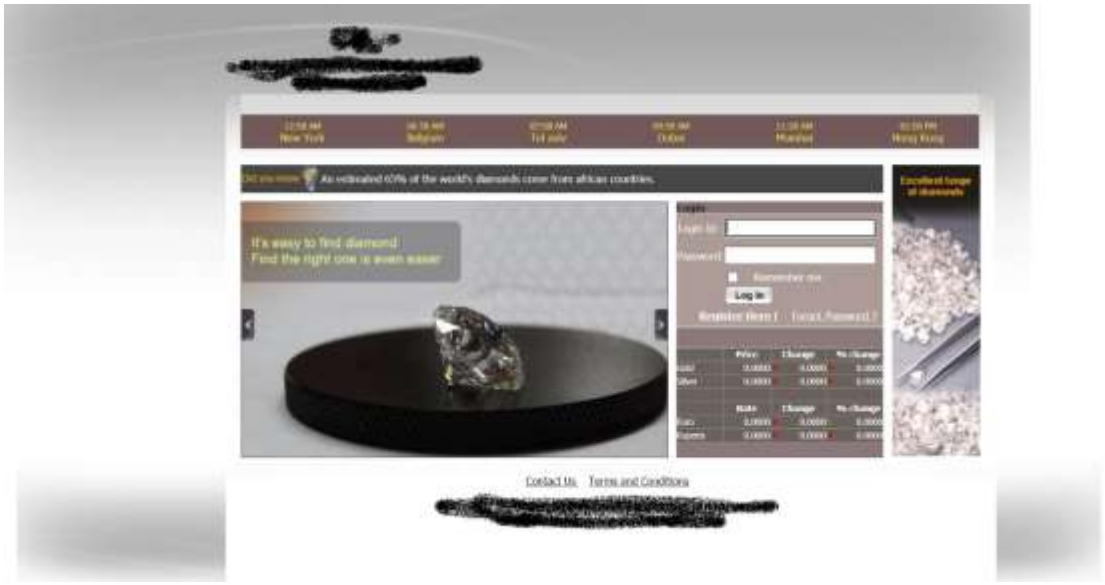
For all other terms, please check google!

The hack:

As I have already said, this hack was conducted because a friend of mine asked me if there was any way that this website could be breached, which I agreed to, as a challenge.

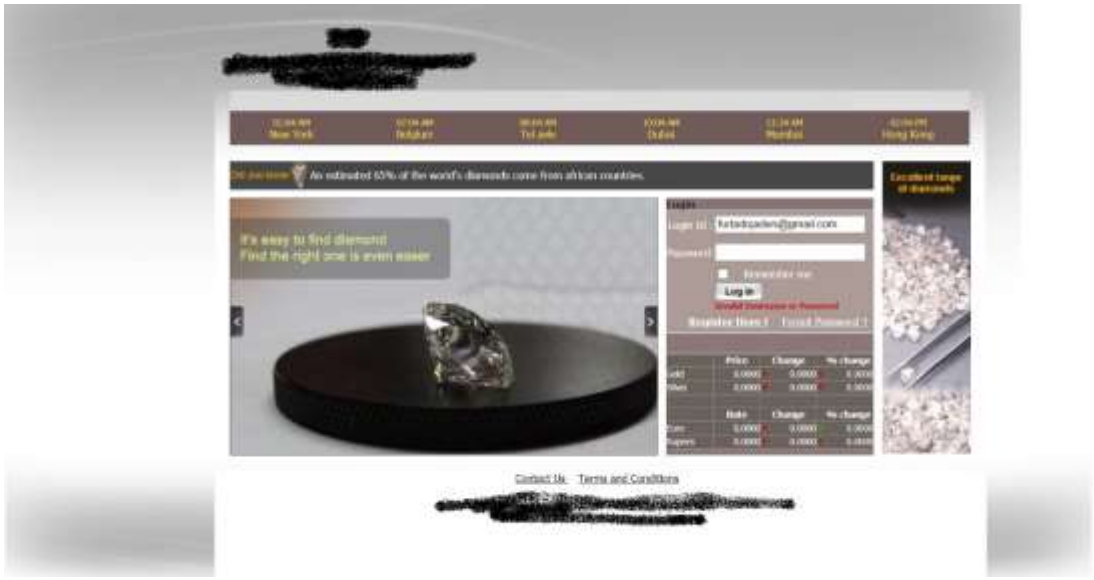
So, I started by analysing the website and all its pages that I could visit.

This is the home/index page of the website.

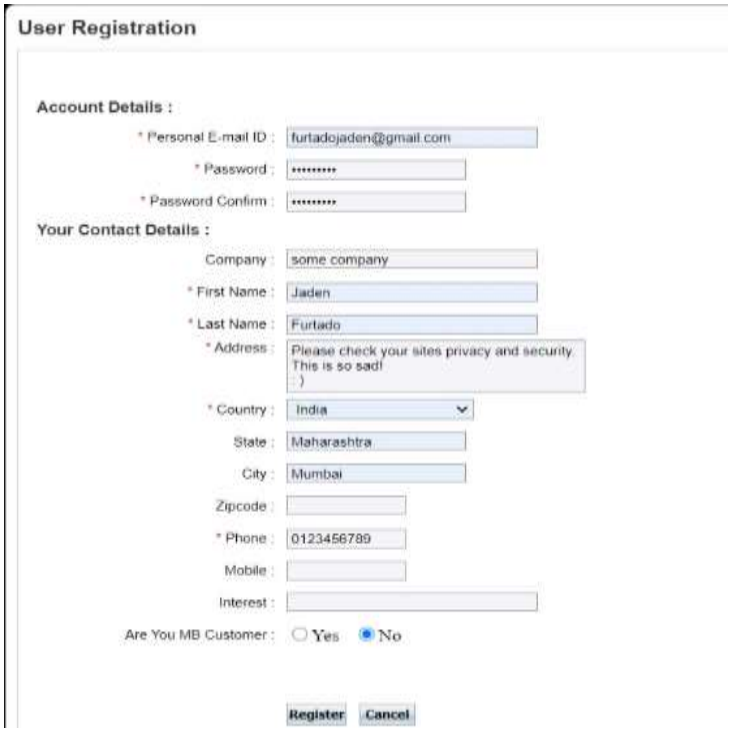


The website is built using the ASP.NET framework. They have used C# in combination with Visual Basic.

When I try and login with an account that I have not registered with, I get an error as follows:



So, I try and register myself.



When you see the details that I have entered, you may say that it's cheeky of me (see the address, phone no. and company name) ;)

But I am doing this to prove a point! This site is accepting any data, without validation. The security is really bad from the offset.

This form is for the admin registration to the site. On registration, I get a page telling me to check my email for a mail containing a link from the admin. On clicking this link, I get verified.

I forgot to take a screenshot of that page, so sorry!

However, the interesting part that I noticed is that, "***an ADMIN HAS TO SEND THE LINK***"! My guess is that this procedure is not automated. They would want to make sure that only an authorised person can grant access to the website. This makes sense, as we are dealing with a precious commodity such as diamonds in this case.

To check if a user is logged in, a site usually sets a session for each user that logs in. The session will contain something that is unique to the individual user, such as email, user_name, user_id, etc. If the session is empty, it means that a user has not logged in.

----------we can express it in psudo code as follows----------

```
If(session_var!=NULL){
        User_is_logged_in;
}
Else{
       User_is_not_logged_in;
}
```

Now that is where things start getting sinister.

>: ->

My guess is that, "***they are storing the username that I just registered in a session variable, without any verification"***, based on the page notifying me to check my mail for the link. **But this is the same session variable that they are also using to check and see if a user is logged in as Admin or not!** So, in theory, if I visit a page that I am not supposed to visit without logging in, I should be granted access by the site.

And that is where the vulnerability lies:

To break it down for someone who is not familiar with web programming, consider these two programmes. In the 1st, you are asked to go to the login page on successfully registration or it sends you a verification email. In either case, no session variable is being set.

----------example of correct code----------

```
//function saves data
Function user_regsitration(user_details){
Save_user_details;
Return True on success or false on failure;
}

//function registers users
Function register_user(){
```

```
    If(user_registration(user_details)){
          Print("successfully registered, please check email for link or go to login page");
     }
     Else{
         Print("An error occurred!");
    }
}
```

However, in their case, they are effectively setting a session variable before login.

----------code with flaw----------

```
//function saves data
Function user_regsitration(user_details){
Save_user_details;
Return True on success or false on failure;
}

//function registers users
Function register_user(){
    Set_user_session_var=user_id;//this is the error, as, user has no authorization, but is logged in!
    //even if this function fails, I have effectively logged in
   If(user_registration(user_details)){
          Print("successfully registered, please check email for link");
     }
     Else{
         Print("An error occurred!");
    }
}
```

They have set the session before I can be verified. And so, I have logged in before I can actually log in!

Let's test this hypothesis:

To prove my hypothesis that I have effectively logged in without actually logging in, I went to the "manage my account" page of the site:

This is what I got.

Although it's not too clear from the Screen Shot, you can see the logout sign on the upper right-hand corner and the update profile and change password fields in the centre of the page of the account options. Which in turn proves that I have logged in!

How did I find this page, when I did not know the names of the pages?

I wanted to see if I can list all the pages that are present in a given directory. In this case, the registration form has a link as follows:

----------This is just a demo, not the real site! ----------

https://www.myWebSite.com~/WebForms/DiaRegistrationForm.aspx

The director therefore is "WebForms".

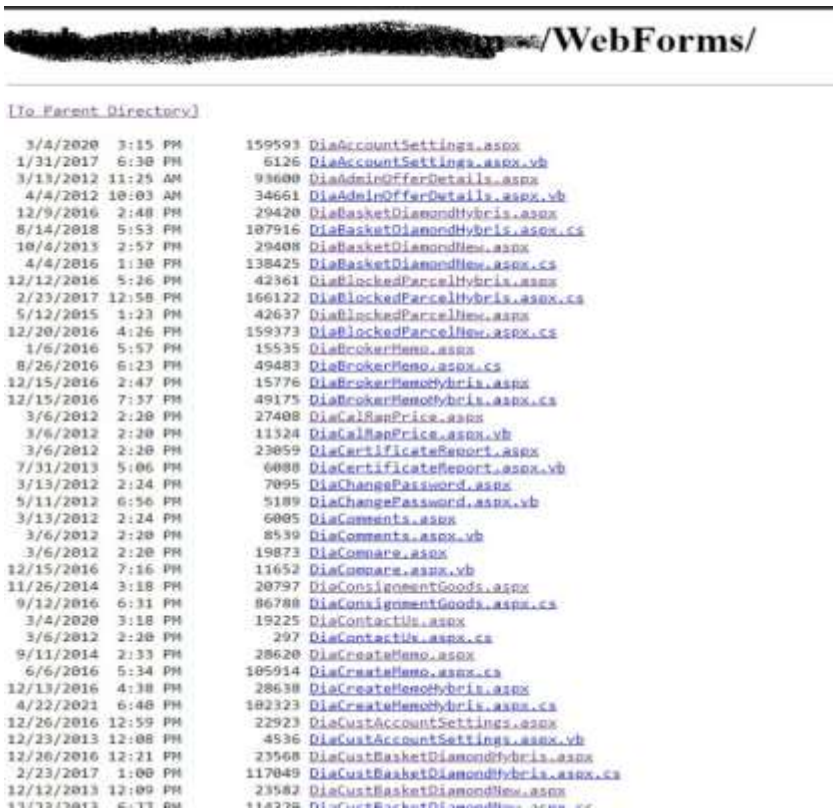
So, I try and see if by chance, they have forgotten to add an index page to any directory.

I entered the link:

----------the link I entered----------

https://www.myWebSite.com~/WebForms/

I hit the jackpot on the 1st try! They have not added the index page in this directory, just as I predicted.



The site has just listed all the pages in the director for me! If you see closely, some of the names of the pages are pretty interesting, especially for an attacker.

I can place/edit/delete orders, items (diamonds, in this case), accounts, etc from their database and so could inflict serious damage on them.

I decided that I would not take any screenshots of the pages/data that I had access to, to protect the company and its customers.

But we haven't finished with this website just yet.

At this point I am in the driver's seat of the company. However, they also have a blind SQL injection flaw!

This took some time for me to find.

I saw that they were sanitizing user inputs before using them. But I eventually found a field, in this case it's "Internal Comment", where you can insert the SQL sleep command and the server runs the SQL code.

You could run pretty much any SQL command, but as it is a blind injection, getting the outputs will be tricky. I have demonstrated an SQL injection on another website before, check that out.

Conclusion:

The deign of the site is not modular.

The user data is directly interacting with SQL database. This however, is dangerous and user data should go through a middleware.

Also, you are supposed to ensure that user is logged in before fetching user data from the backend. This is an additional step to prevent a breach in data.