SECURITY

FINAL PROJECT REPORT

Team-Members:

1-Amr Hatem

2-Youssef El-hady

3-Yassin Bayoumi

4-Karim Labib

5-Mohamed Magdy

6-Ahmed Al-khatib

Project:

Implementing Twitter with security measurements, this project is implemented from scratch we didn’t use open source projects.

Technologies used:

PHP, Mysql, Jquery, HTML, CSS.

General features:

1-user can register in the website.

2-user can login using credential.

3-user can edit his password, profile picture, cover picture, Bio, status, privacy status, Country And Website if any.

4-user can search for another users in the search bar.

5-user can view other user’s profile

6-user can like any tweet if logged in.

7-user can unlike any tweet if logged in.

8-user can retweet any tweet if logged in.

9-user can comment any tweet if logged in.

10-user can delete his comments.

11-user can delete his tweets.

12-user can view other user’s profile if and only if this profile is public or if he’s following a private profile.

13-user can follow/unfollow other users.

14-user can logout.

15-user can mention another user.

16-user can write a public/private Tweets.

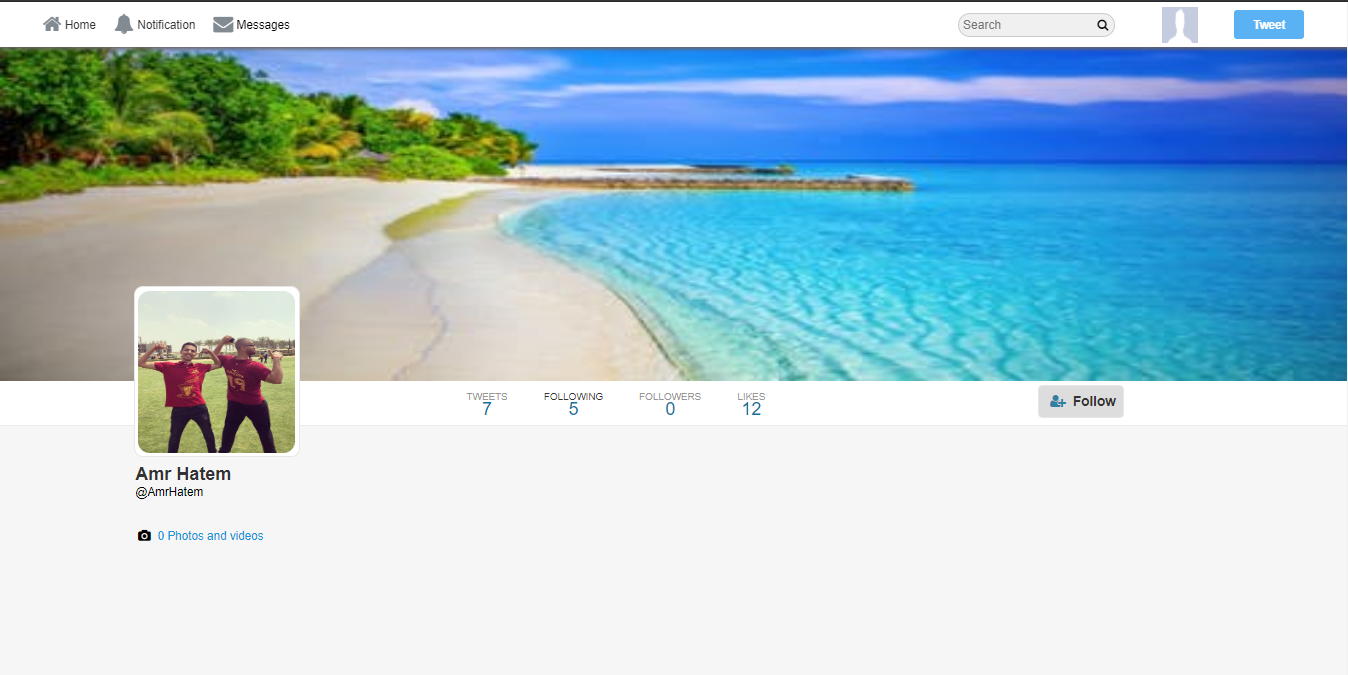
17-user can view who to follow accounts.

Security features:

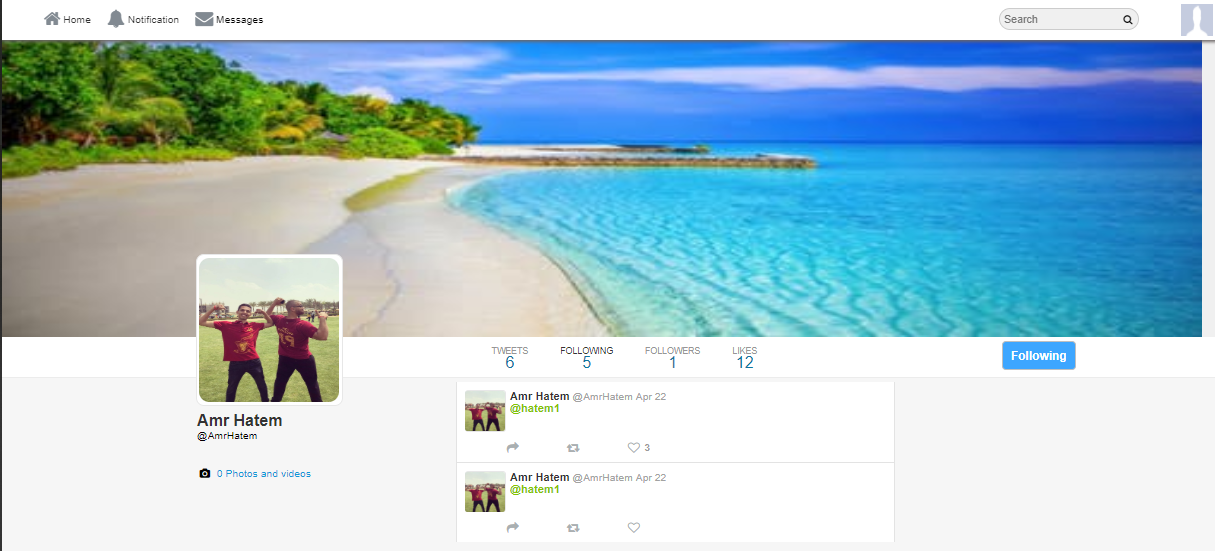
1-Authentication: Each user can logged to system if and only if he/she is authorized to be able to use system’s feature (like-unlike-retweet-comment-tweet)

2-Access control: user can only do things he’s allowed to do Ex: user can only edit/delete/view what the system or other users allow him to, we implemented private/public accounts so that private account’s posts cannot be seen unless this private account is followed by the user viewing this private account.

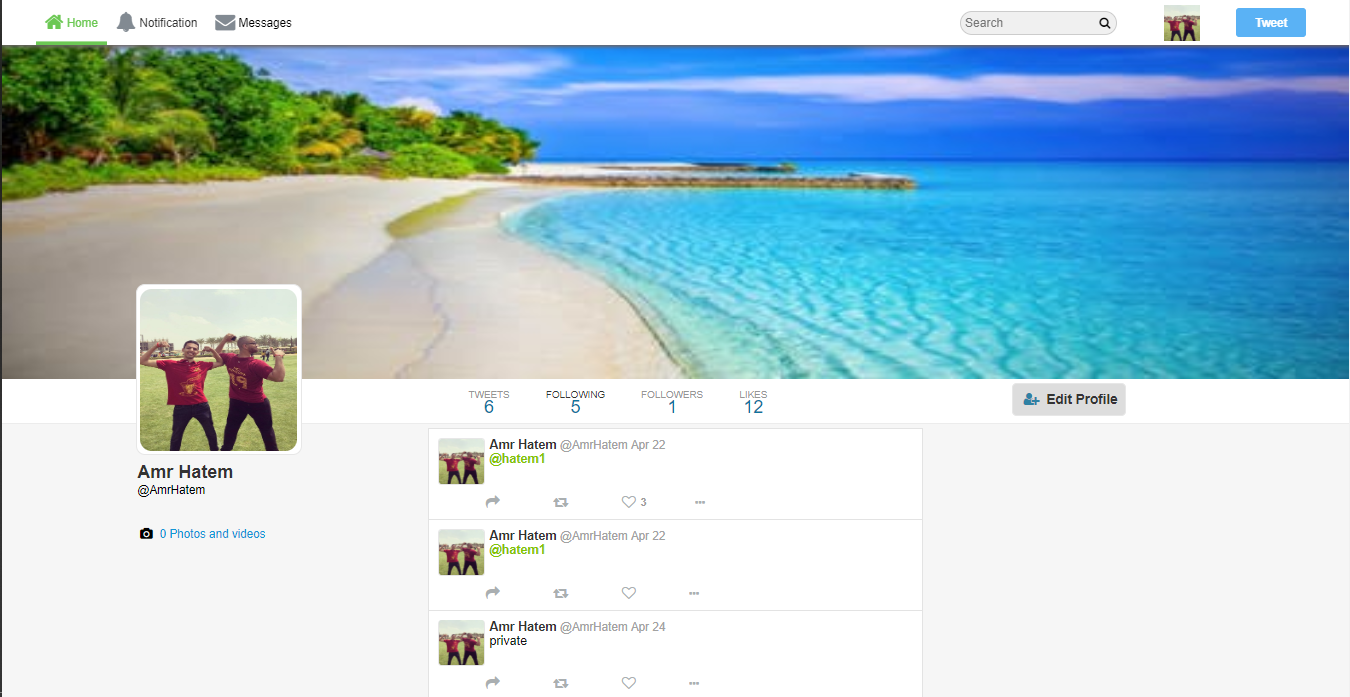
So here’s a private account. And a user tries to view this private account(he can see nothing).



And here After he managed to follow this private account.



We also implemented private posts that is visible only by the user posting this tweet even the account is public as seen in the next image (notice we are now viewing Amr Hatem as Amr hatem not other user so we can now see private post).



3-Encryption and Hashing: We stored the password of users encrypted in our databases, we used salt. Were each user has a unique random generated salt concatenated to their original password then hashing this combination using md5. By this we ensure that same passwords in will have different hashes in database which will make it harder for any harder for any hacker if happen to have the database to crack all passwords.

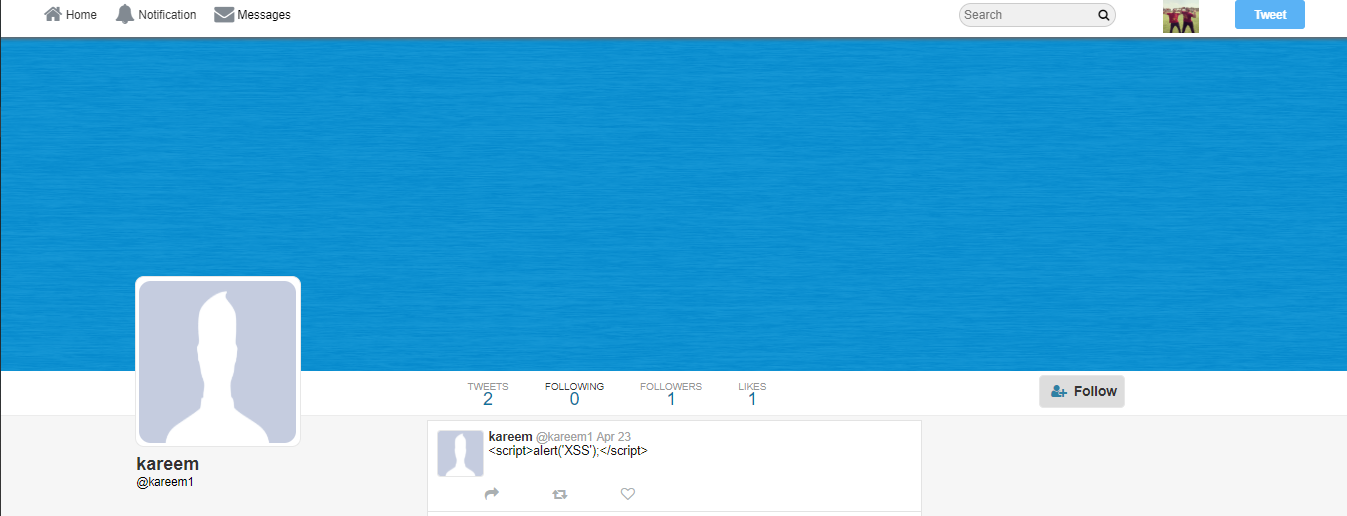
4-Access Tokens: We used Sessions in order to maintain user’s authentication while he’s using the website so that he doesn’t have to login each time he browse any service.

5-SQL Injection prevention:

In order to protect our website from SQL injection we used Mysql prepared statements, we made sure that any SQL query done in our website is sanitizing the input also we bind parameters so that you cannot treat parameter as query or combine queries together which is common used by attackers.

6-XSS Prevention:

By escaping/filtering html characters, we used the function htmlspecialcharacters() that take html code convert it to html entities so that no html or js can be injected in our website. This method is used for any possible text field where the user can write input.



Resources:

1-<https://www.acunetix.com/blog/articles/prevent-sql-injection-vulnerabilities-in-php-applications/>

2- <https://stackoverflow.com/questions/1996122/how-to-prevent-xss-with-html-php>