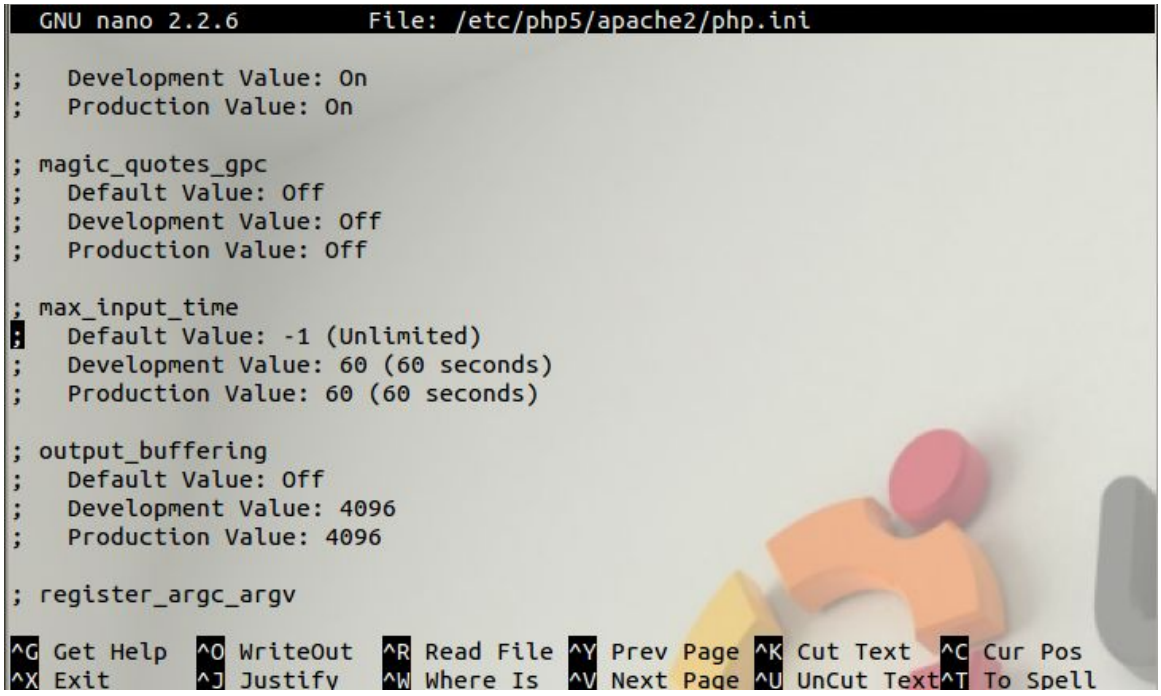
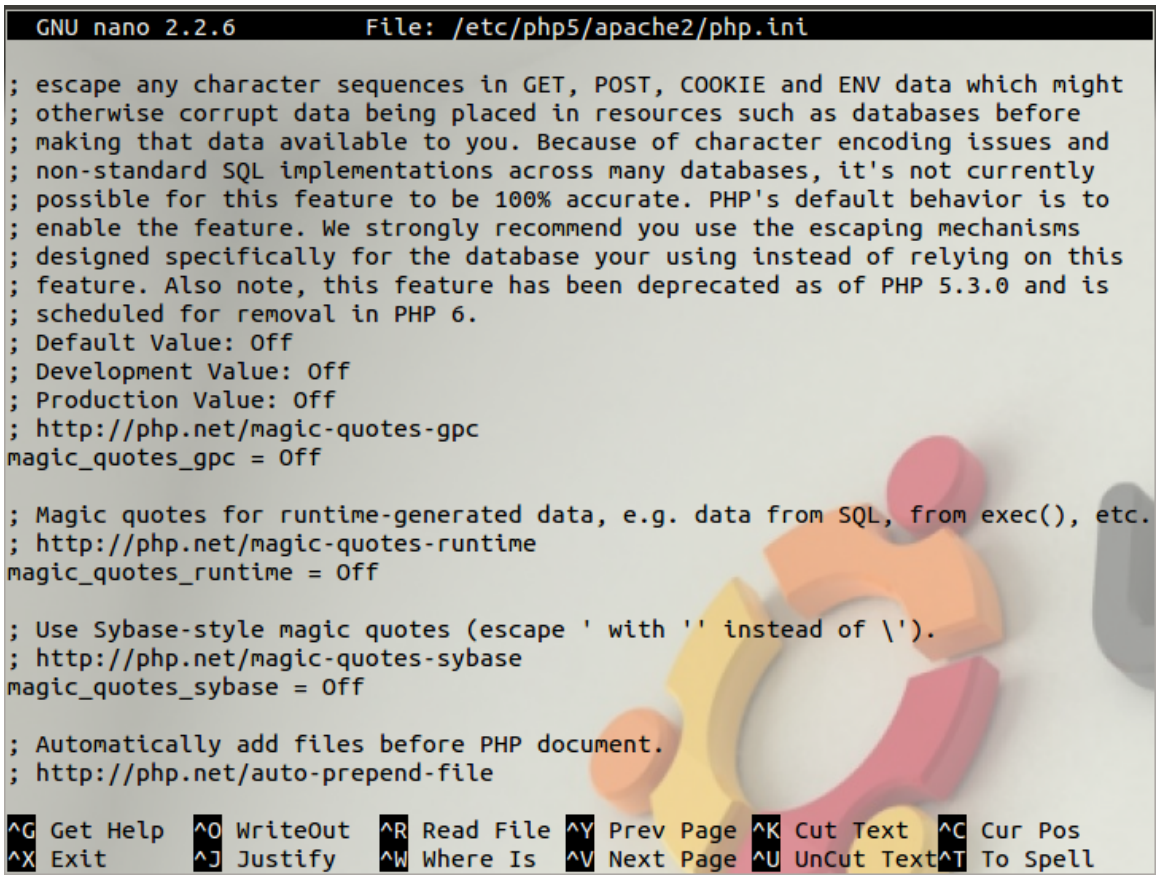
Alfredo Ceballos

https://github.com/ace231/CS380-EX6

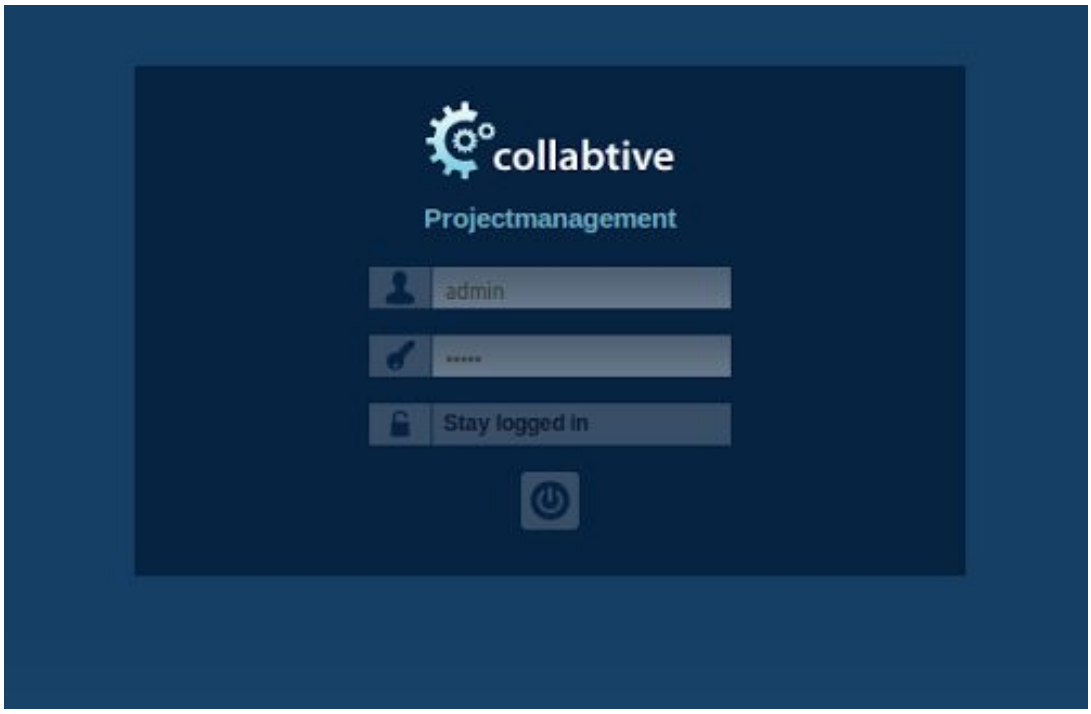
Setup:

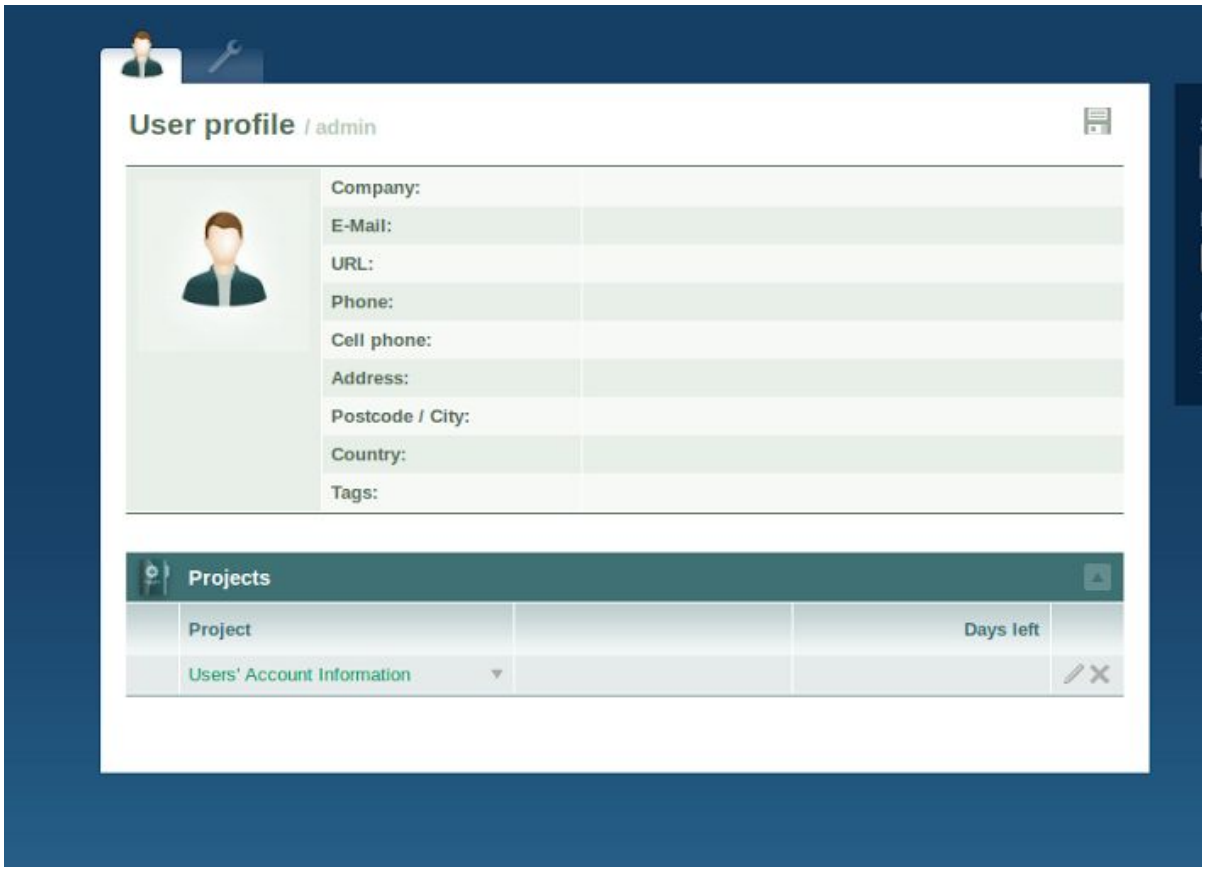




Magic_quotes_gpc value switched to Off throughout the php.ini file.

Ran Apache command restart server and logged in as user admin with password admin





Account page of admin user

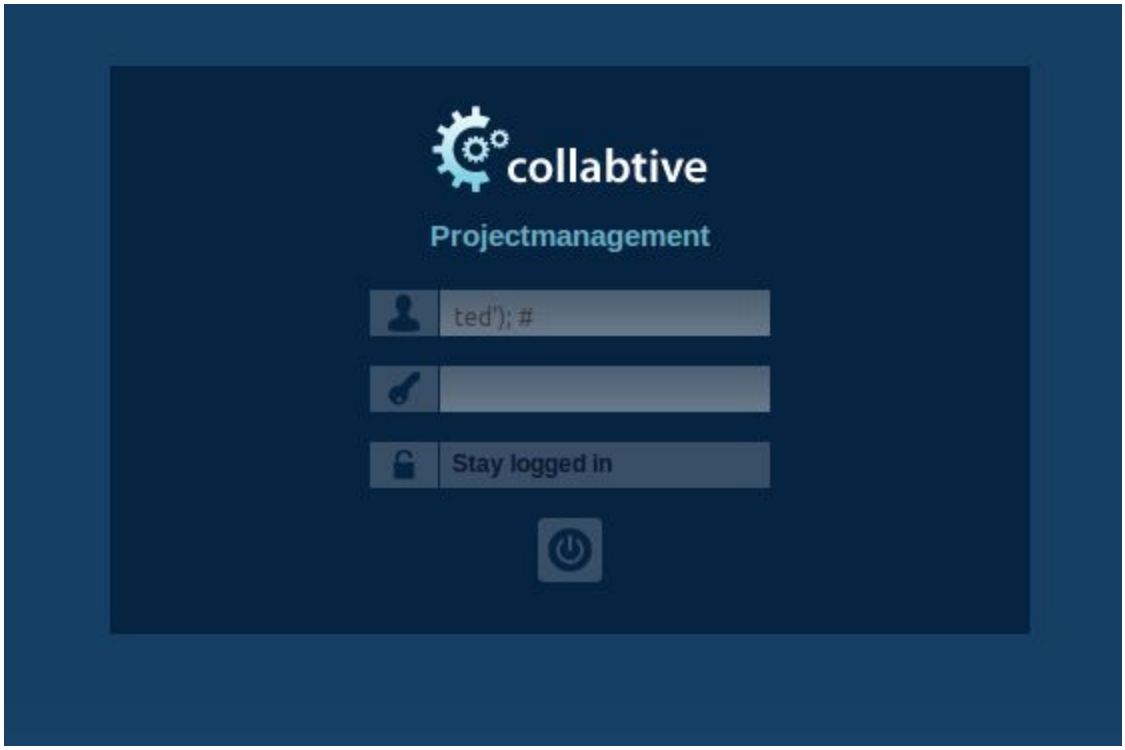
SQL Injection in User Authentication:

The most important line in the login code has to be `WHERE (name = '$user' OR email = '$user') AND pass = '$pass'");`

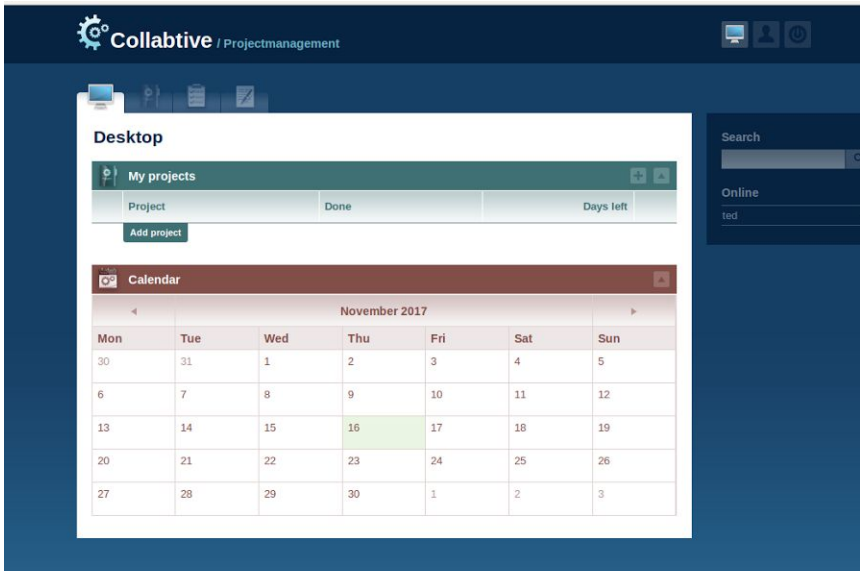
The exploit can be triggered by inputting <username>'); #

What happens is the PHP code will treat the input as actual code since it is not sanitizing or running any sort of check to prevent outside code from being injected into the query. When a valid username is used, the SQL query becomes `WHERE (name = '<username>'); # OR email = '$user') AND pass = '$pass'");` everything from # to the end of the line is ignored as a comment. The query will fetch the record of the inputted username without even needing a password. The record will be seen to exist and the login proceeds.

Here it is working for the username "ted"

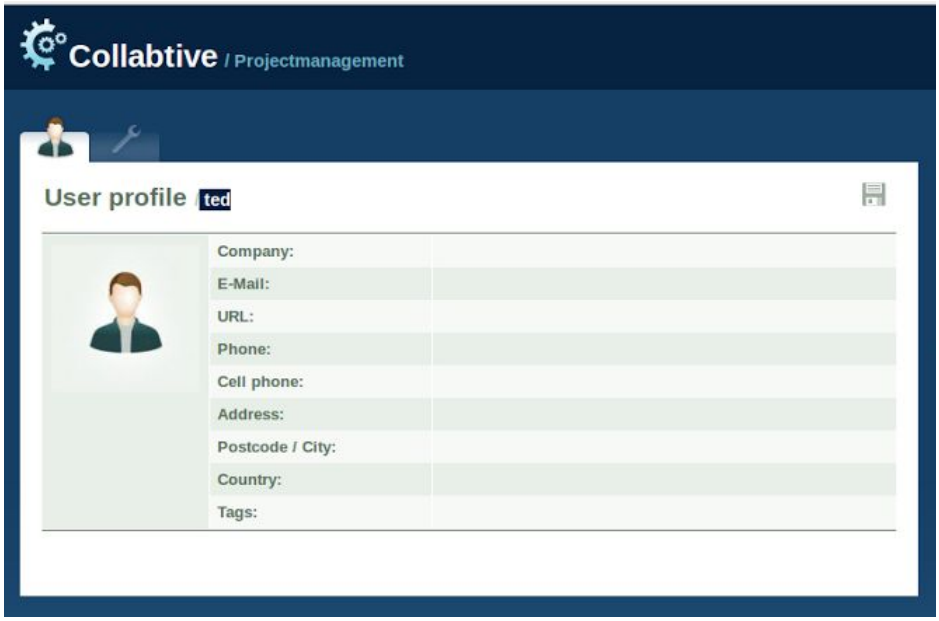


Query becomes `WHERE (name = 'ted'); #` (Remember the rest becomes a comment)

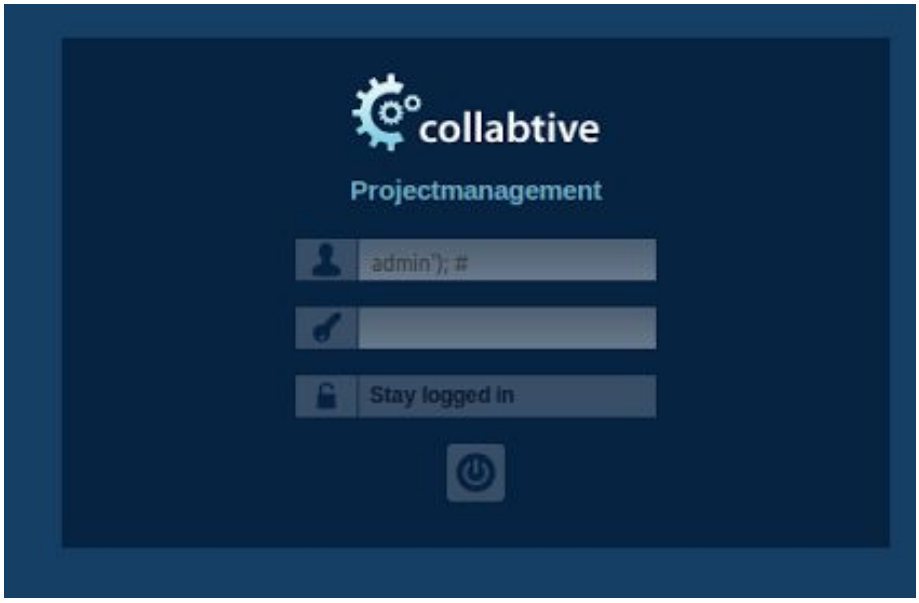


Login was allowed

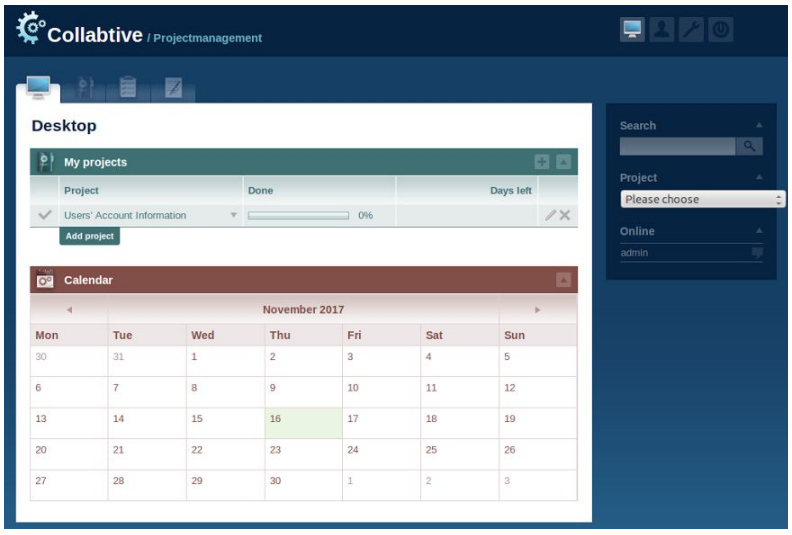
More proof



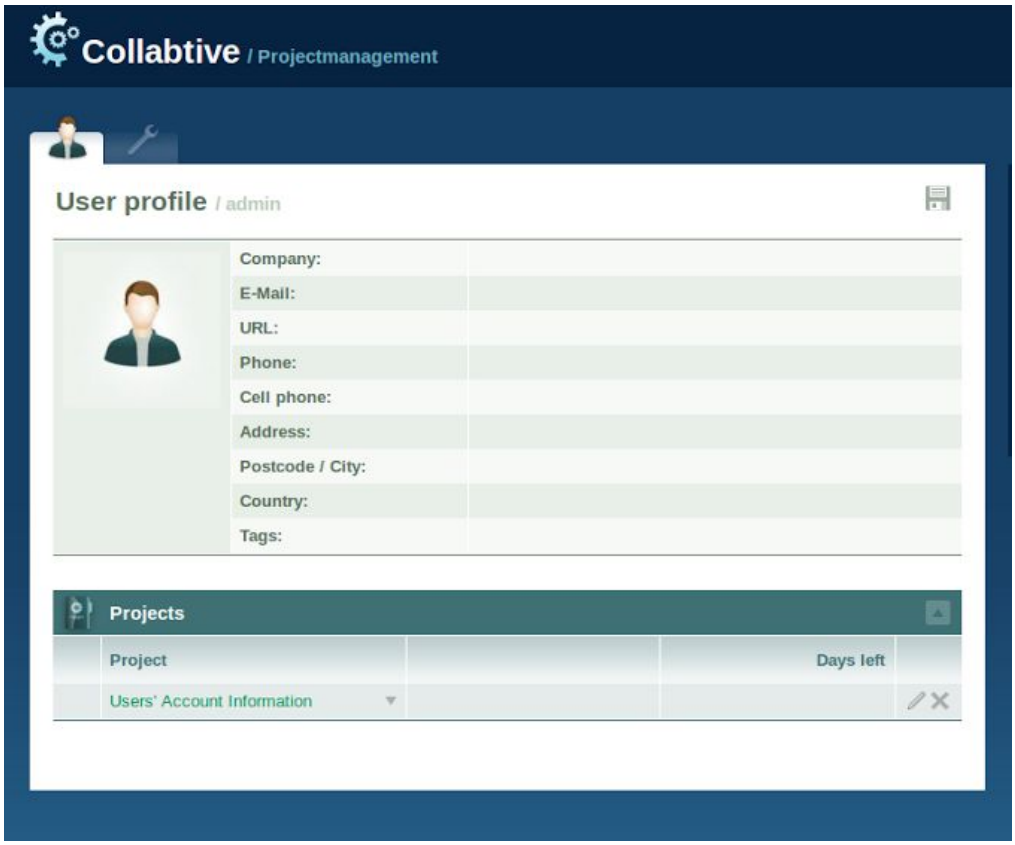
The same logic works with the username “admin”



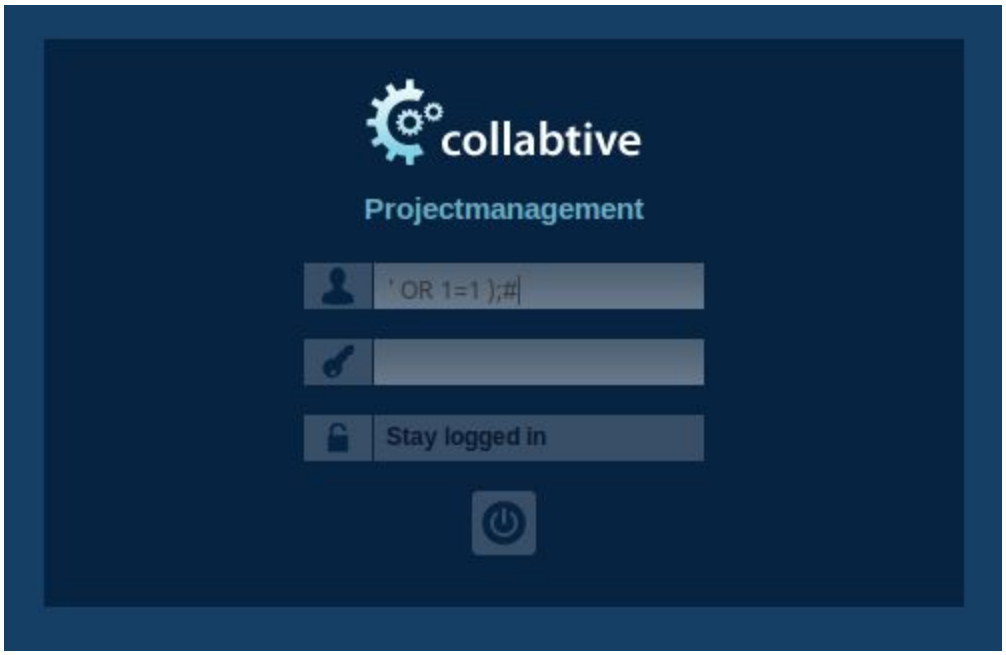
Query becomes `WHERE (name = 'admin'); #` (This example shows a more dangerous scenario)



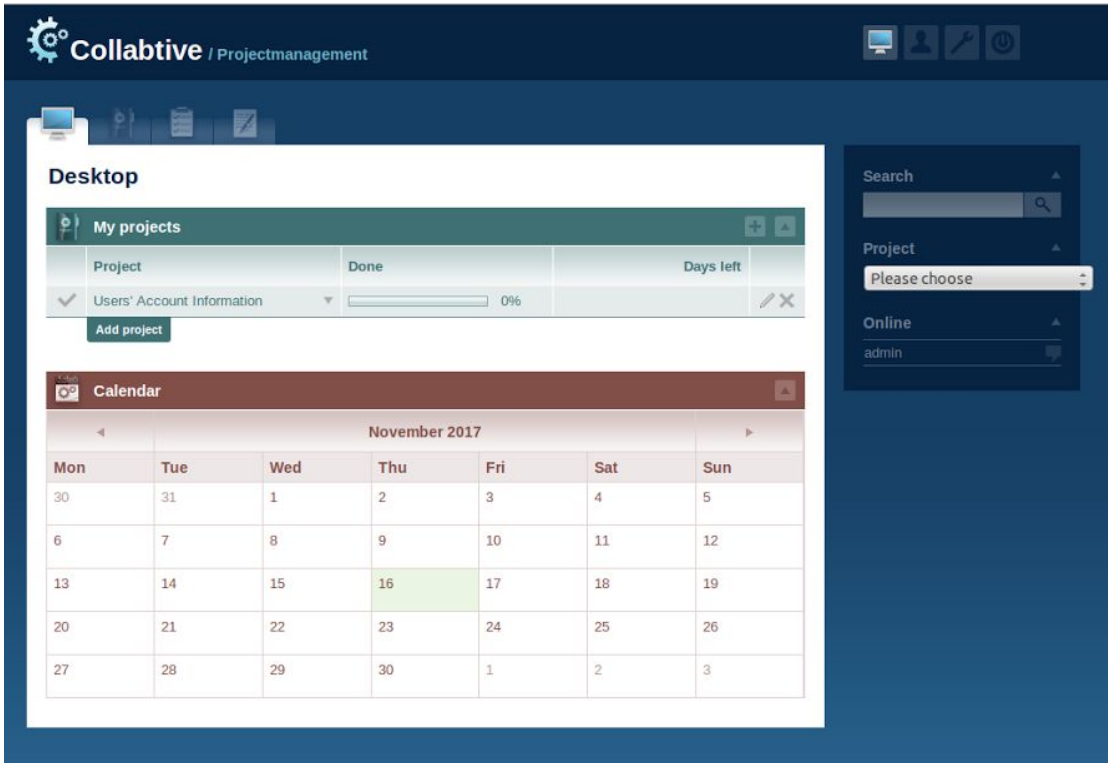
Now logged in as an admin user

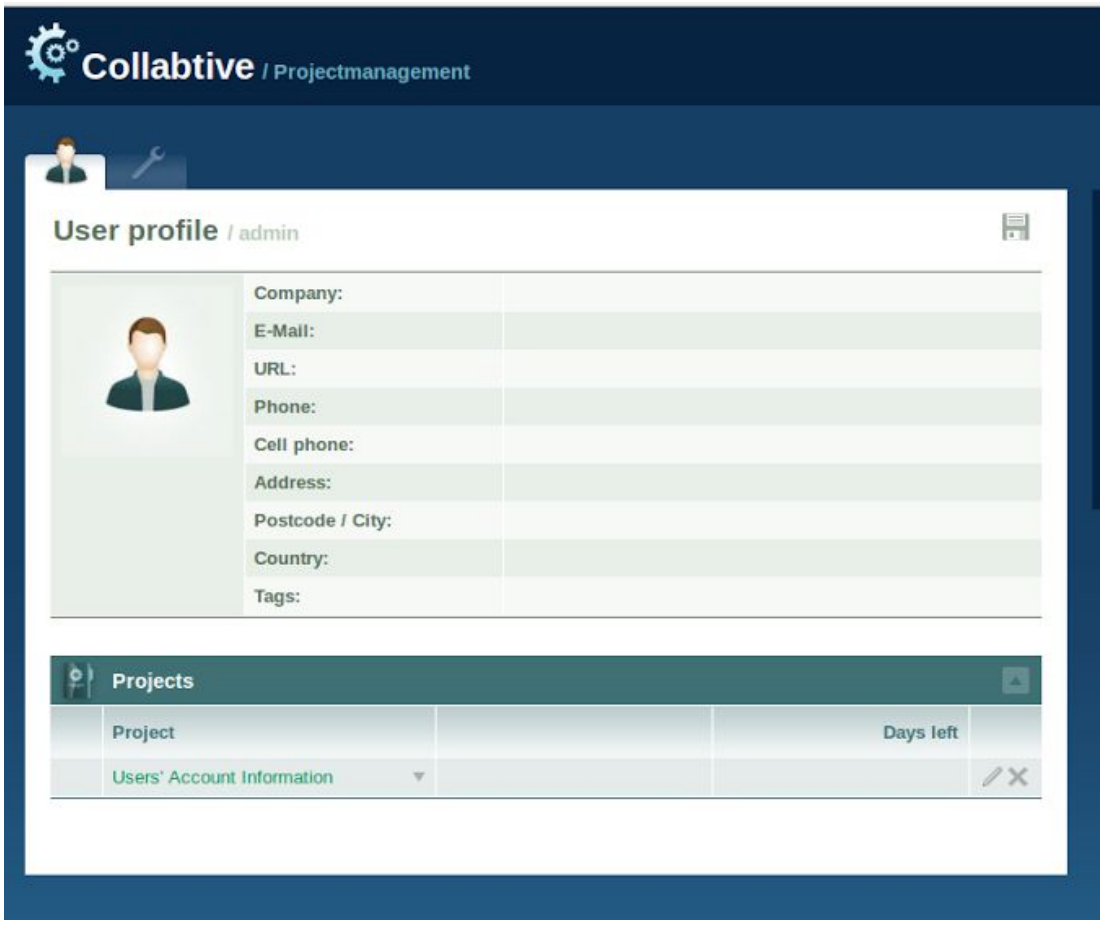


These examples worked only because the inputted usernames actually existed. Would it be possible to login into the website without providing a username at all? The answer is yes.



Here the query becomes `WHERE (name = '' OR 1=1 );# OR email = '$user') AND pass = '$pass'");` and again remember, because of the pound sign/hashtag the only valid part of the query becomes `WHERE (name = '' OR 1=1 );` this returns true no matter what, skipping the login page's user check and allowing the login process to continue… As an admin sadly enough





And there it is, SQL may make managing data and tables easy, but if user input is not sanitized and checked properly, it opens the door to rather simple yet potentially catastrophic code injections.